Realistic Real-time Sky Dome Rendering in Gran Turismo 7

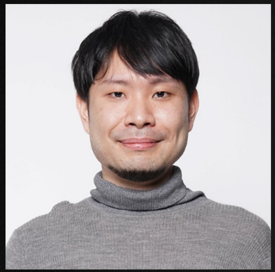
GDC 2023
Polyphony Digital Inc.

Kentaro Suzuki kentaro.d.suzuki@sony.com
Kenichiro Yasutomi kenichiro.yasutomi@sony.com

Hello everyone, let's begin our talk, "Realistic Real-time Sky Dome Rendering in Gran Turismo 7"

I hope you to enjoy it!

Who?



Kentaro Suzuki

Polyphony Digital Inc.
Lead Graphics Engineer



Kenichiro Yasutomi

Polyphony Digital Inc.
Technical Artist

POLYPHONY™ DIGITAL

I'm Kenichiro Yasutomi, a technical artist in Polyphony Digital, and my co-speaker is Kentaro Suzuki, a Lead Graphic Engineer.

Let's get start.

Agenda

- Introduction

- Physics of Sky
- Simulation of Sky
- Real Earth Atmosphere
- Skysim --- Our Sky Renderer

- Run-time Sky Rendering Method
- Run-time Cloud Rendering Method
- Summary

POLYPHONY™ DIGITAL

This is the agenda for today's talk.

Introduction

Let me give you an overview first.

Introduction

Gran Turismo 7



- Driving Simulator for PlayStation®4/5
- More advanced car physics
- More than 400 car models
- Photorealistic visuals

- Dynamic time and weather



POLYPHONY™ DIGITAL

Gran Turismo is a driving simulator that celebrated its 25th anniversary this year and is also a racing game featuring realistic physics and graphics.



(video)

The latest Gran Turismo 7 released in March 2022, features dynamic time and weather changes, creating for a more realistic driving experience.



In-Game Rendering in Gran Turismo 7

Since the time and weather were dynamic,



In-Game Rendering in Gran Turismo 7

so the sky had to be dynamic as well.

The decision was made early on,



we started to develop a simulation of the sky in 2019.

Introduction

Features

- Measurement-based detailed atmosphere model
- Offline sky renderer for physically accurate scatter simulation
- LUT-based approach to use offline rendered images in real-time

These are the features of our implementation:

Measurement-based detailed atmosphere model,
The Offline sky renderer for physically accurate scatter simulation,
and LUT-based approach to use offline rendered images in real-time.

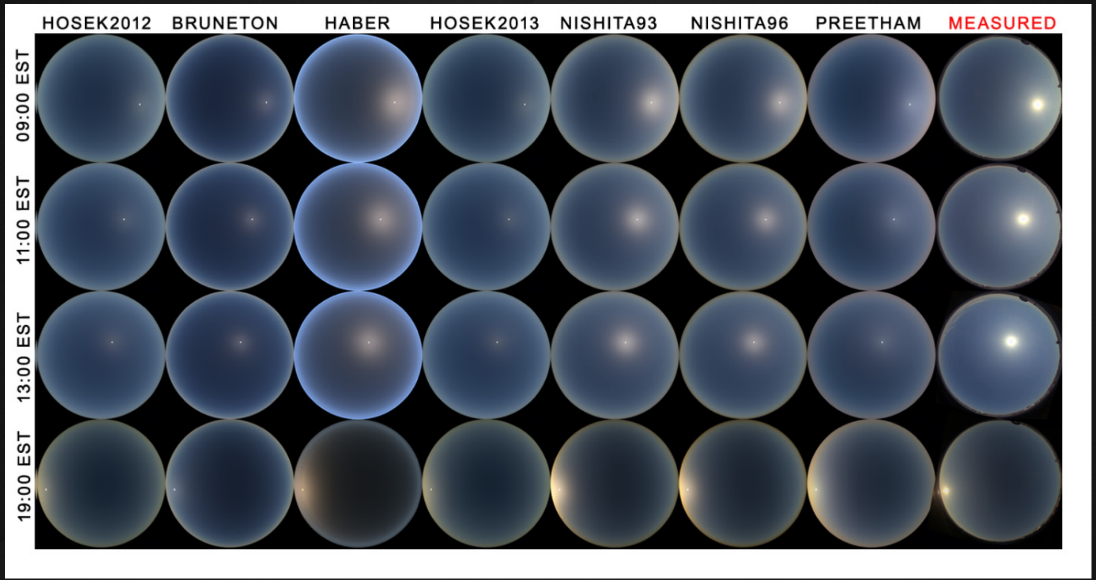
Introduction

Motivation

First, we did some research on existing simulations:

Motivation

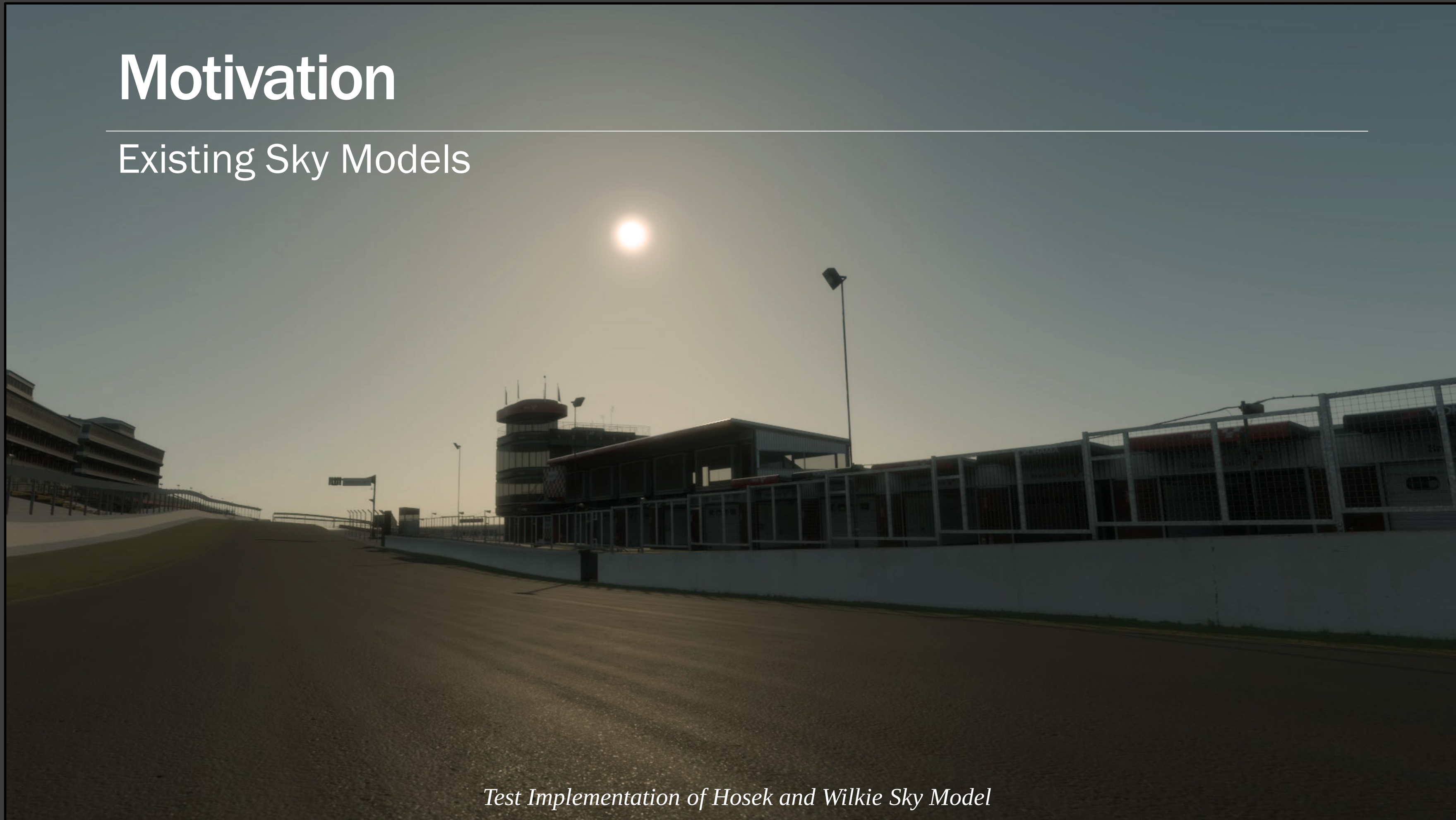
Existing Sky Models



A Framework for the Experimental Comparison of Solar and Skydome Illumination [Kider et.al. 2014]

POLYPHONY™ DIGITAL

Nishita, Hosek, etcetera.



Test Implementation of Hosek and Wilkie Sky Model

We were not very happy with them.

They look a bit dull and hazy as if there was a thin veil on them,



especially the sunset was far from satisfactory.

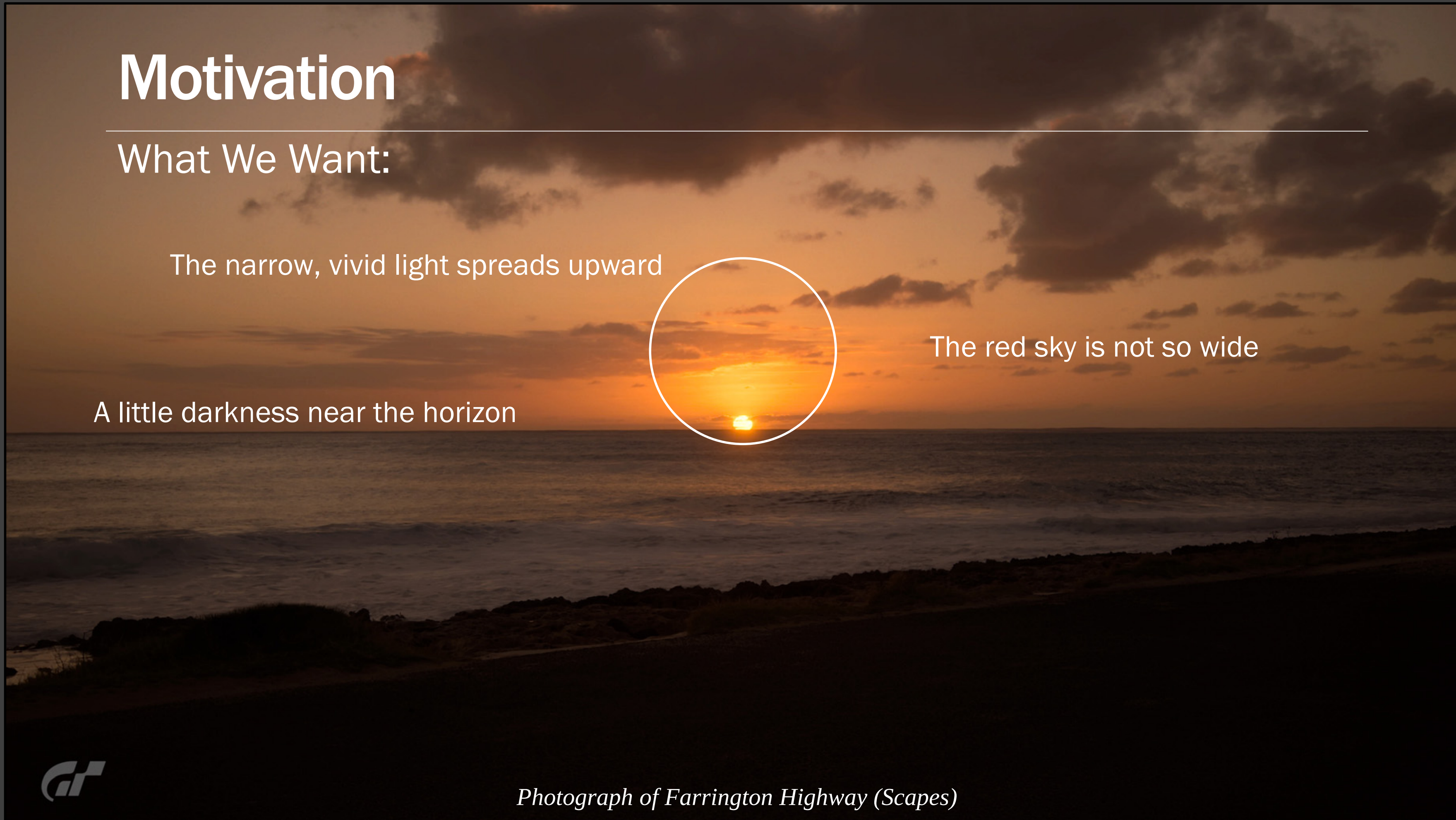
Motivation

The Sunset Sky



Photograph of Farrington Highway (Scapes)

The real sunset has more colors and complex contrast around the sun.



Photograph of Farrington Highway (Scapes)

So, what's the problem?

It would be a matter of accuracy but we had no idea what was critical and whether it could be improved.

First, we surveyed many science papers to know reality.

Agenda

- Introduction

- Physics of Sky
- Simulation of Sky
- Real Earth Atmosphere
- Skysim --- Our Sky Renderer

- Run-time Sky Rendering Method
- Run-time Cloud Rendering Method
- Summary

POLYPHONY™ DIGITAL

Before we get into the main topic,

Physics of the Sky

let's take a brief look at the premise.

Physics of the Sky

What is the Sky Light?

POLYPHONY™ DIGITAL

What is the sky light?

What is the Sky Light?

Light from the Atmosphere

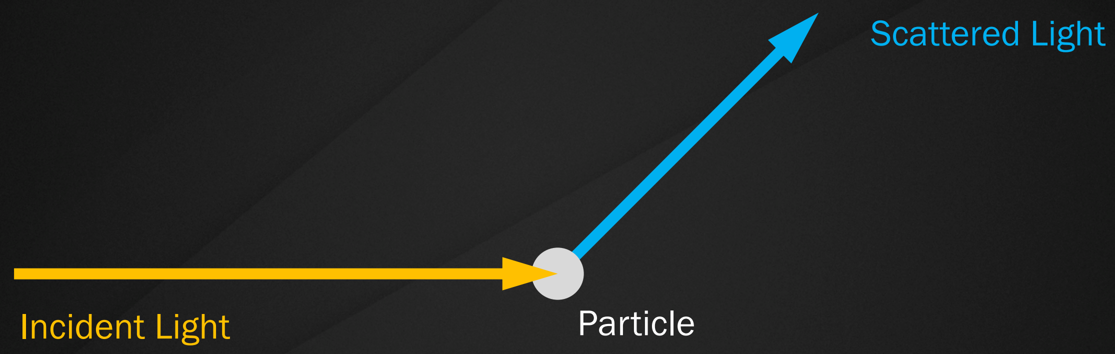


Photograph of Finnmark (Scapes)

It's the light from the atmosphere illuminated by the sun.

What is the Sky Light?

Light Scattering

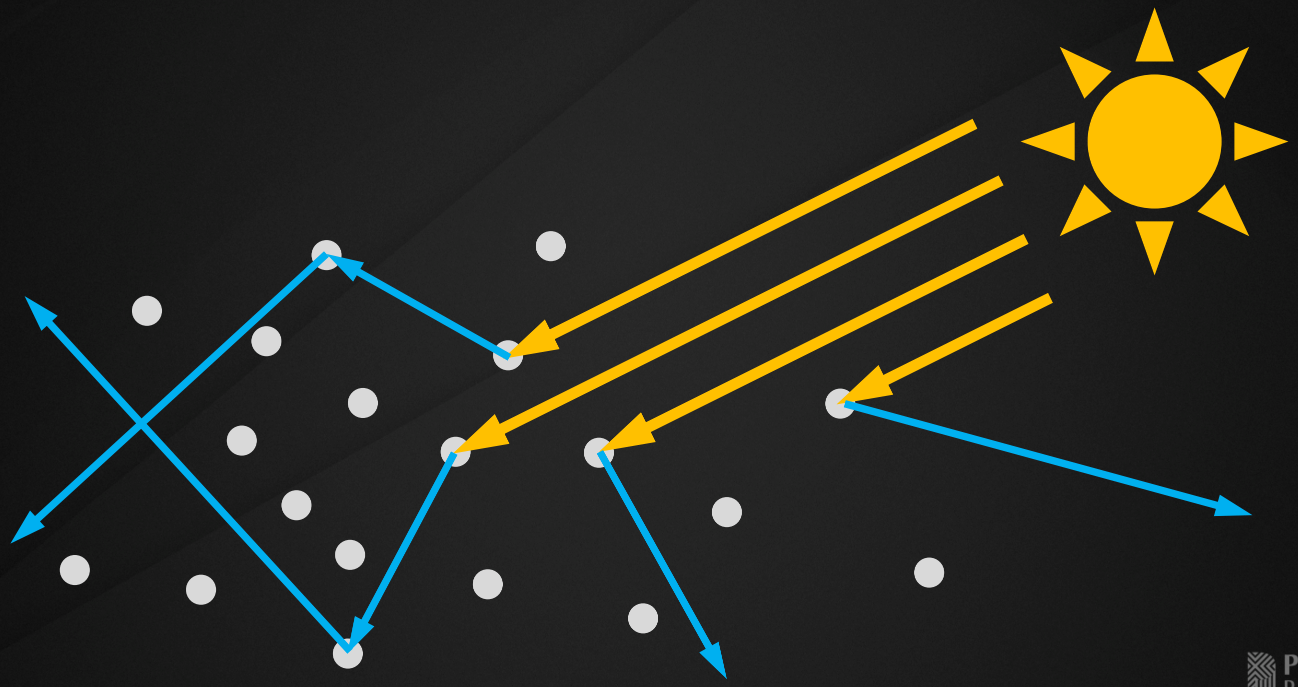


Light interacts with small particles changing the direction of propagation from the original one.

This phenomenon is known as light scattering.

What is the Sky Light?

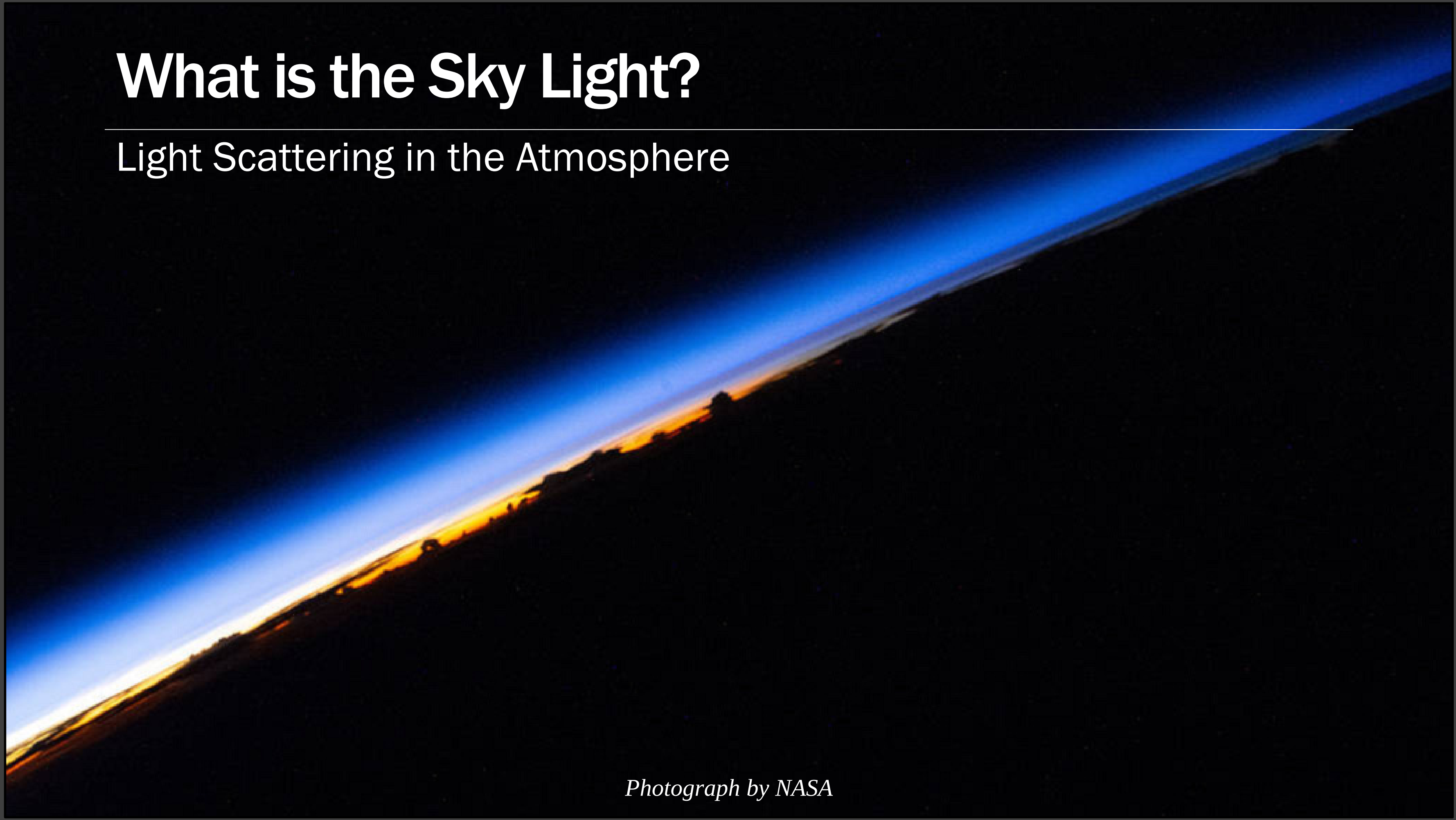
Light Scattering in the Atmosphere



There are countless particles floating in the atmosphere that scatter sunlight in different directions.

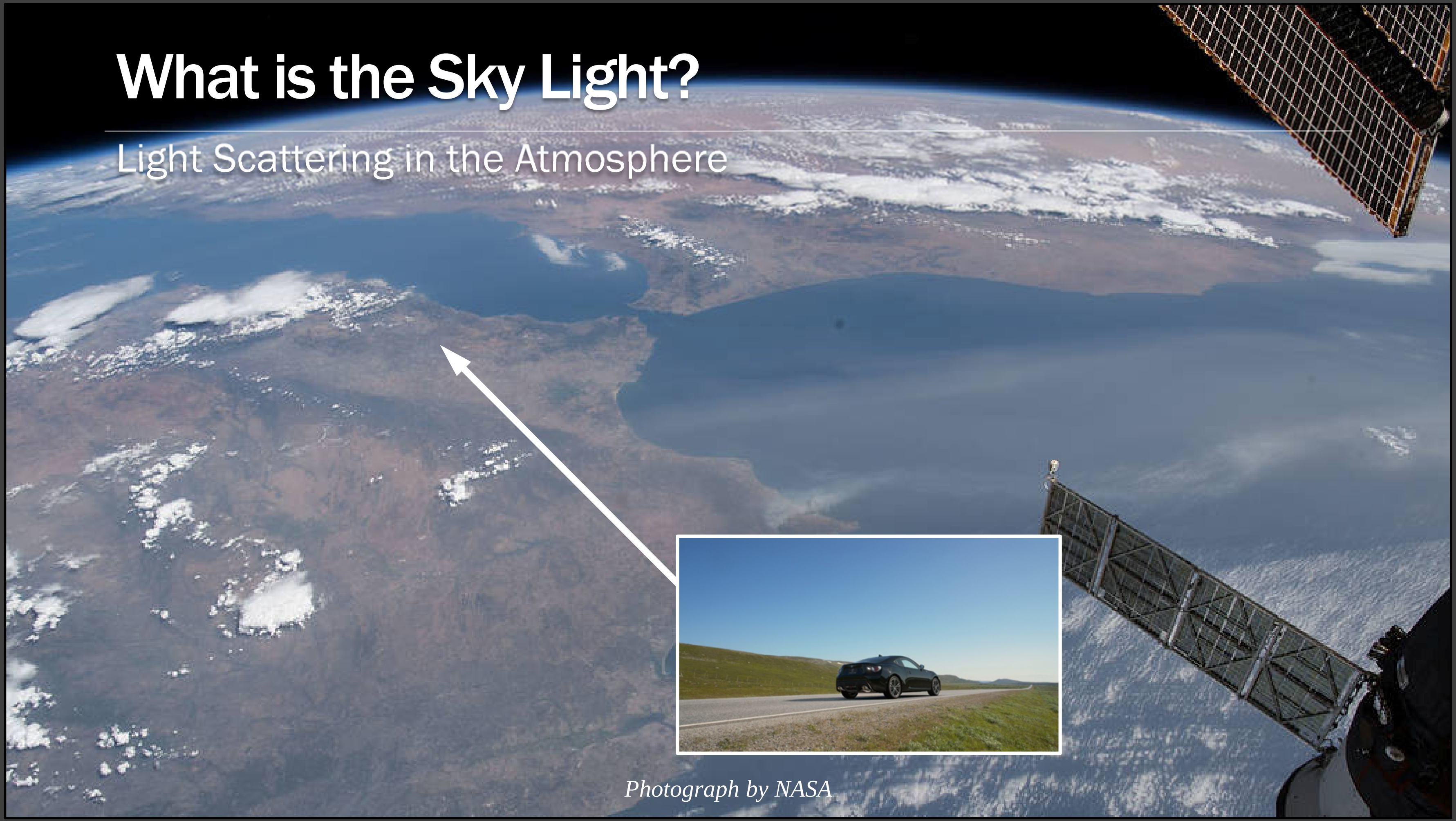
What is the Sky Light?

Light Scattering in the Atmosphere



Photograph by NASA

The atmosphere that scatters the sunlight appears to glow,



We are inside the glowing atmosphere, looking up from the ground.

Physics of the Sky

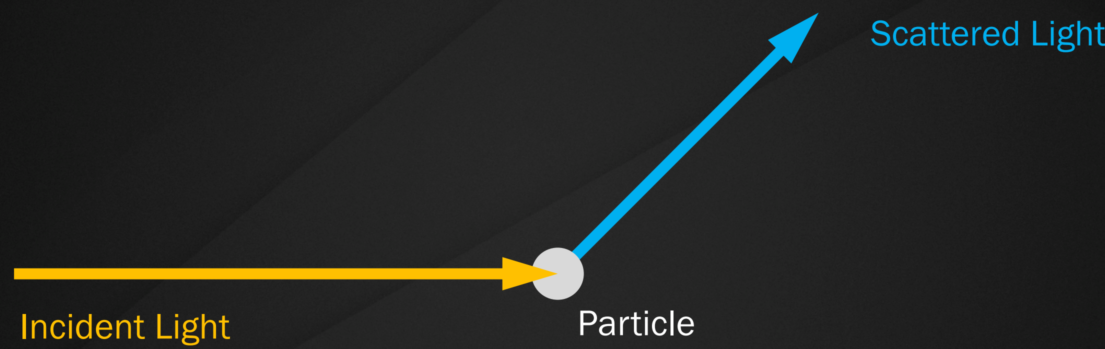
Law of Light Scattering

POLYPHONY™ DIGITAL

Next, let me introduce the physical laws of scattering.

Law of Light Scattering

The Light Scattering



The most important fact is that scattering changes the direction of light,

Law of Light Scattering

The Light Absorbing

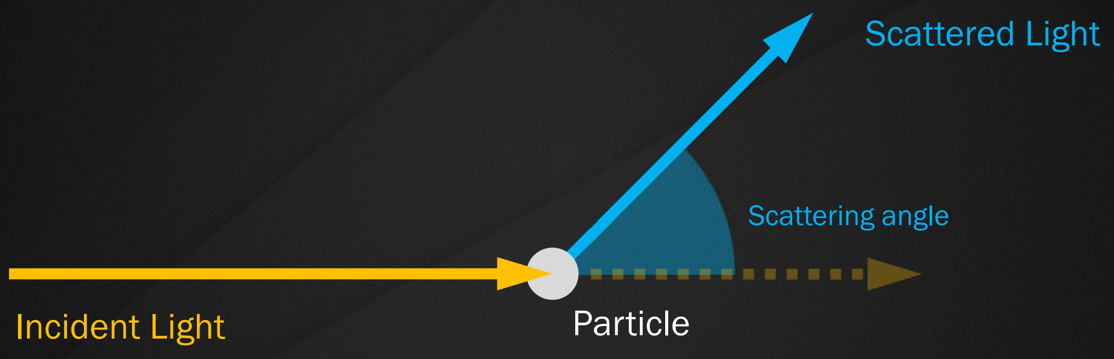


POLYPHONY DIGITAL

But they not only scatter they can also simply absorb.

Law of Light Scattering

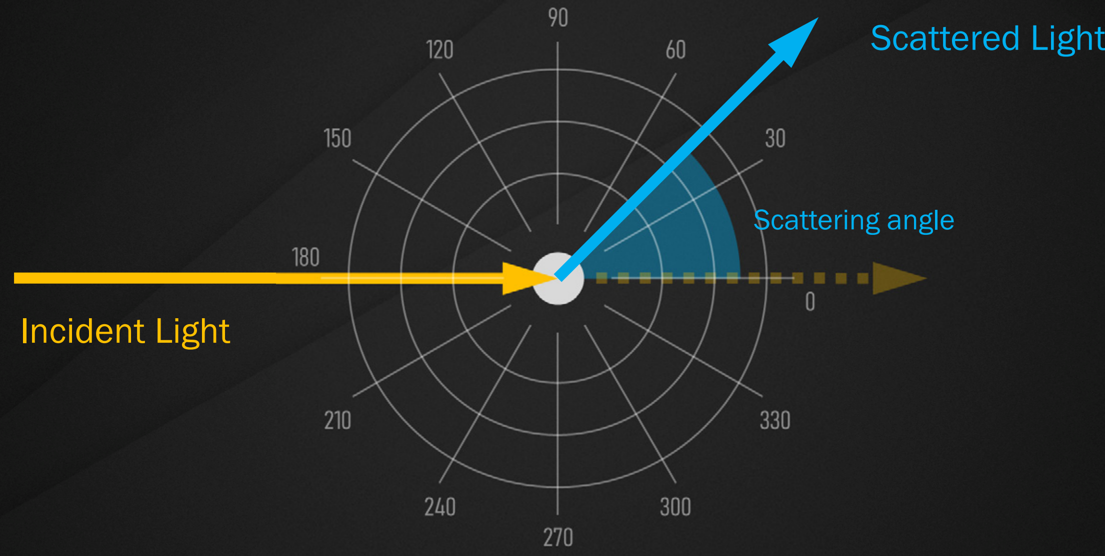
The Phase Function



And there are a law governs the direction in which the scattered light is emitted.

Law of Light Scattering

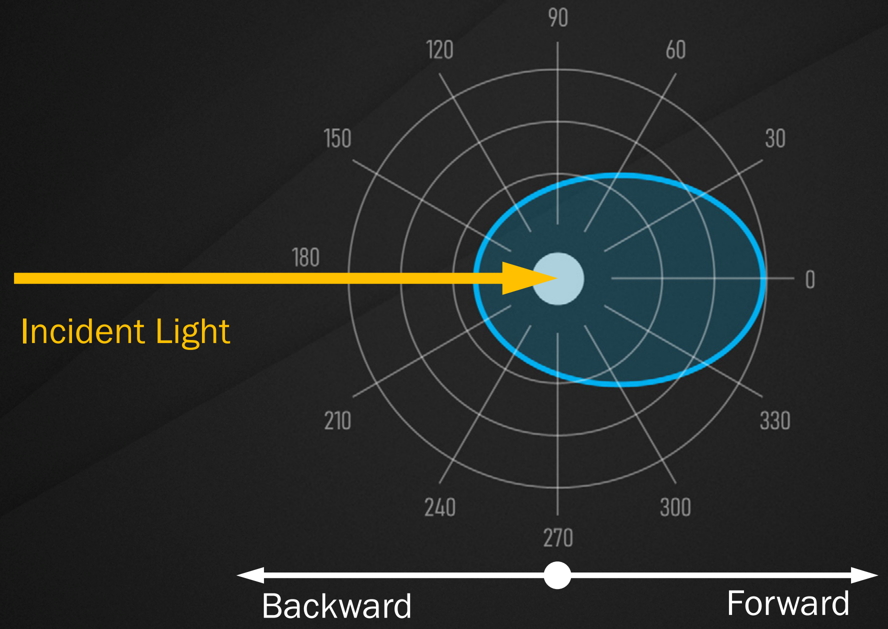
The Phase Function



This is given by the phase function, which describes the angular distribution of the scattered light.

Law of Light Scattering

The Phase Function



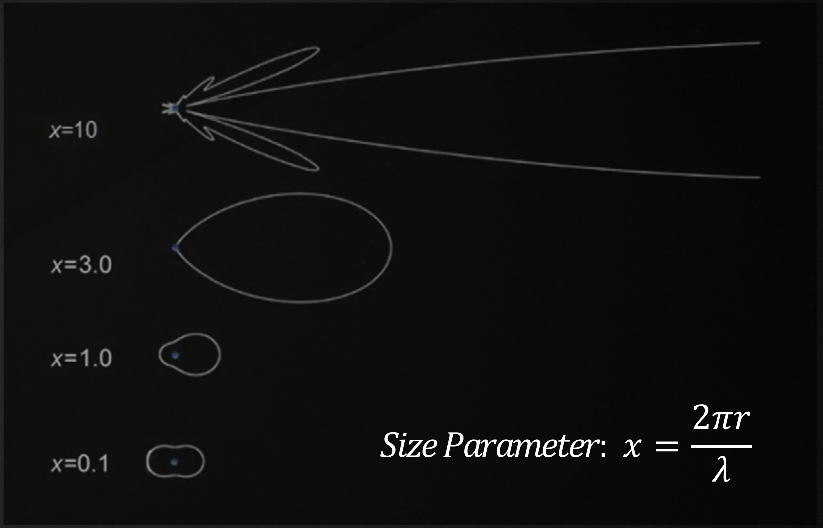
POLYPHONY DIGITAL

When the phase function is like this, the scattering is more forward and less backward from the incident direction.

Law of Light Scattering

The Phase Function

- Particle Refractive Index
- Particle Size
- Light Wavelength



$$\textit{Size Parameter}: \ x = \frac{2\pi r}{\lambda}$$

Figure: W.Brune

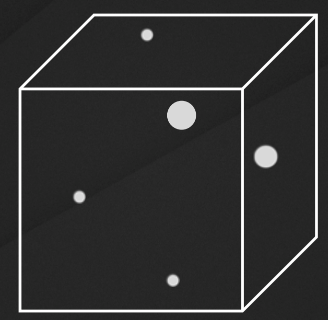
POLYPHONY DIGITAL

The phase function is physically determined by the refractive index of the particle, the particle size, and the wavelength of the light.

In general, the larger particle size, the more the scatter in the forward direction.

Law of Light Scattering

Scattering Parameters

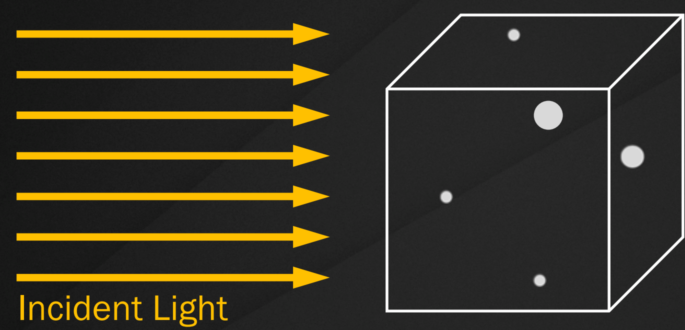


Then, let’s consider at the macro perspective, we should think about the density or type of particles in a volume.

Suppose there are some particles in a volume.

Law of Light Scattering

Scattering Parameters

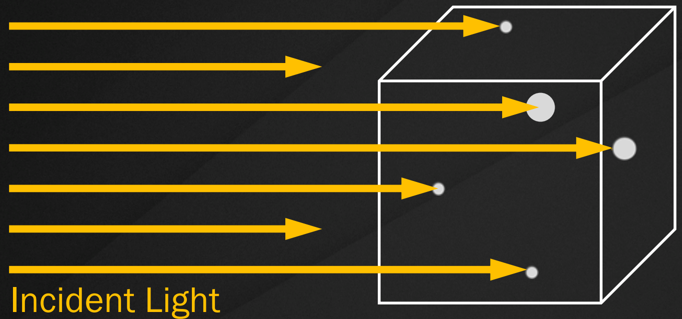


POLYPHONY DIGITAL

When the light enters this volume,

Law of Light Scattering

Scattering Parameters

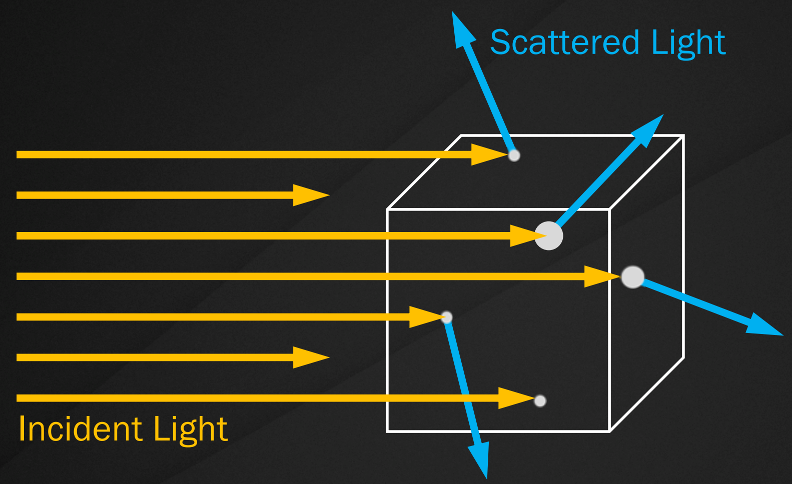


POLYPHONY™ DIGITAL

the light that interacts with particles is [*] scattered according to each phase function,

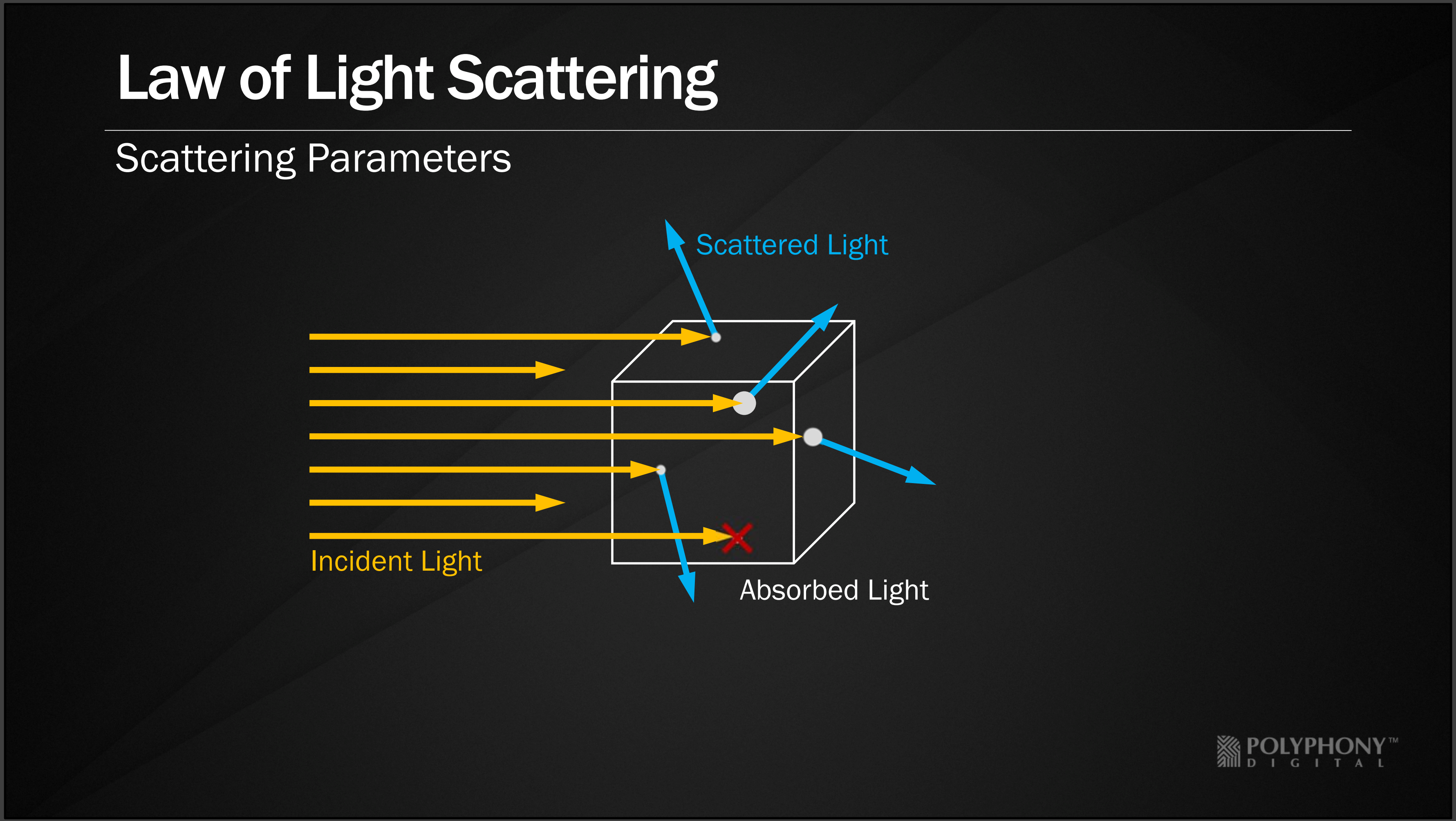
Law of Light Scattering

Scattering Parameters



POLYPHONY™ DIGITAL

the light that interacts with particles is [fwd] scattered according to each phase function of particles, or [fwd] absorbed,

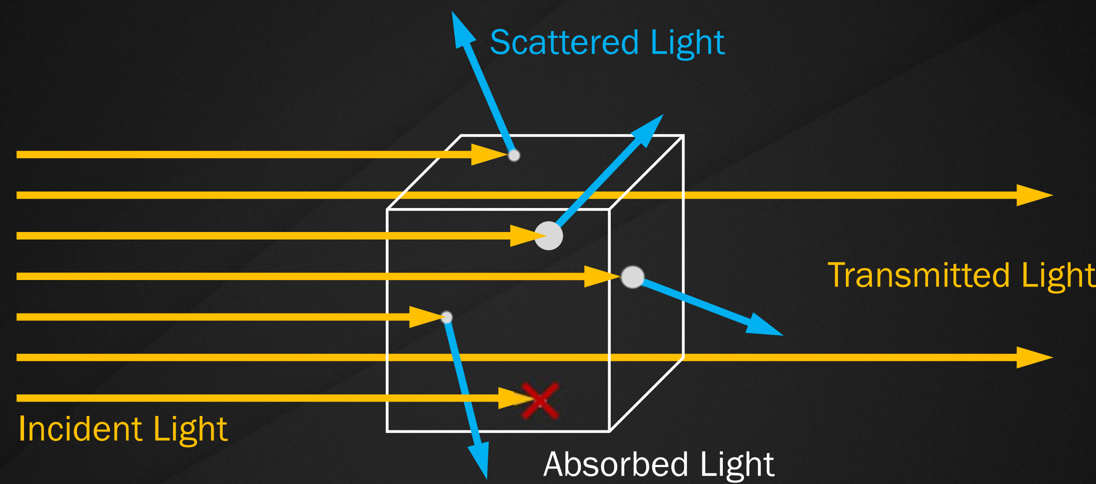


or [fwd] absorbed,

while the light that does not interact with any of them [fwd] simply passes through.

Law of Light Scattering

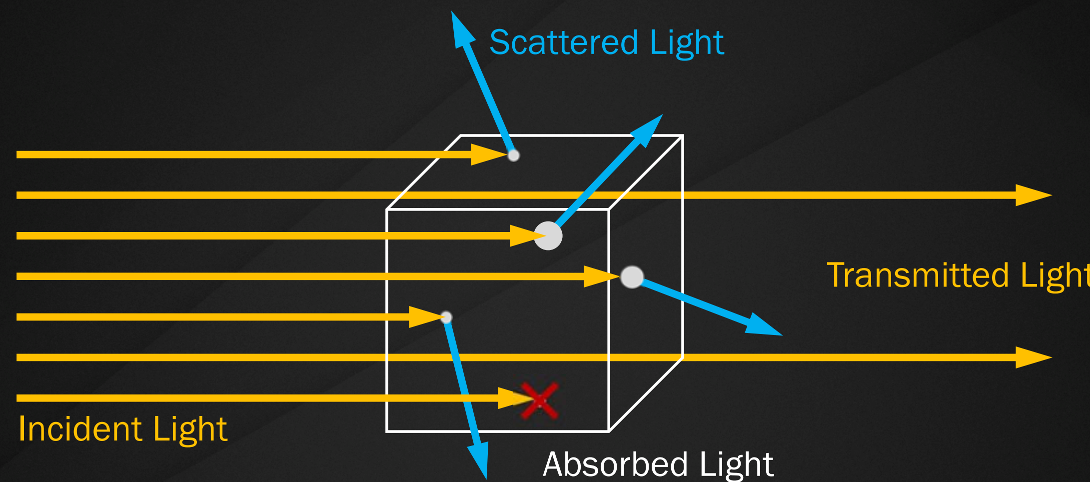
Scattering Parameters



simply passes through.

Law of Light Scattering

Scattering Parameters



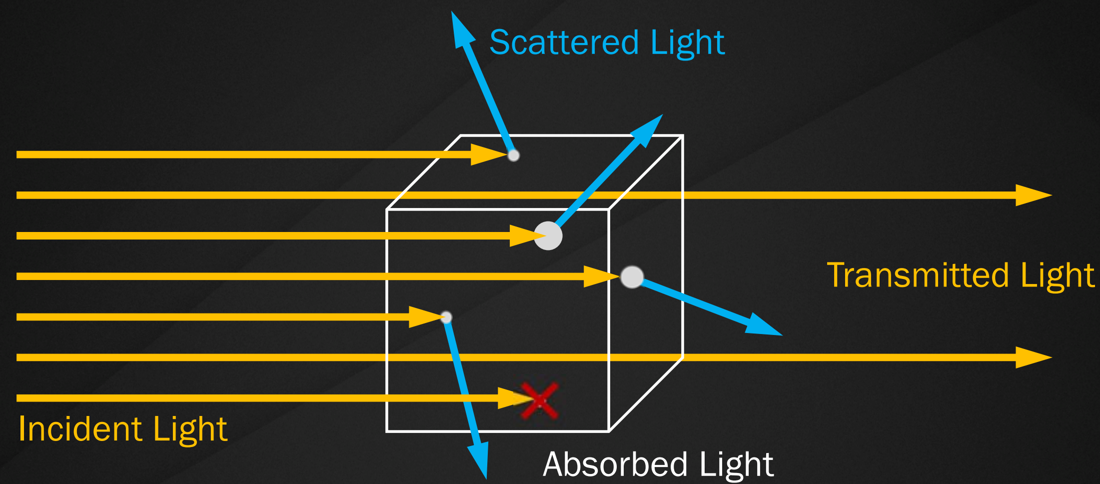
Incident Light = *Scattered Light* + *Absorbed Light* + *Transmitted Light*

POLYPHONY™ DIGITAL

And a conservation law of energy is there.

Law of Light Scattering

Scattering Parameters



Scattering Coefficient [1/m]

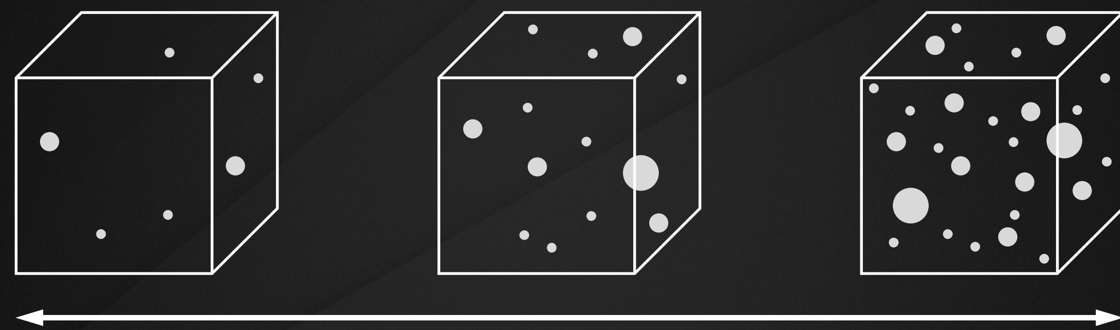
Absorption Coefficient [1/m]

So, they are described as intensity coefficients per unit distance that light travels;

how much it is scattered, how much it is absorbed, and how much it is transmitted that remaining.

Law of Light Scattering

Particle Density



Less Scattering
Less Absorbing
More Transmission

More Scattering
More Absorbing
Less Transmission

The denser the particles are, the more likely they are to interact with the light;

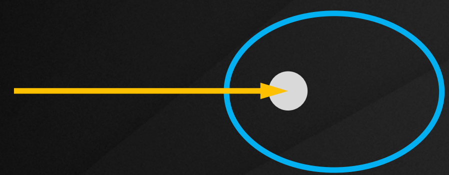
so scattering and absorbing will increase, transmission will decrease.

Law of Light Scattering

Scattering vs Reflection

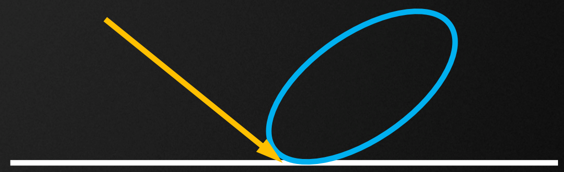
Scattering

Scattering coef. * phase function



Reflection

BRDF



POLYPHONY™ DIGITAL

Combining the scattering coefficient and the phase function is similar to BRDF you should know well.

Physics of the Sky

Rayleigh and Mie

POLYPHONY™ DIGITAL

The next are prominent theories.

Rayleigh and Mie

The Two Scattering Approximations

Rayleigh Scattering

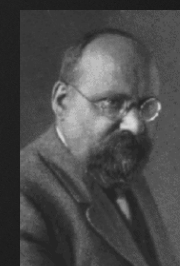
Law of light scattering by particles smaller enough than the **wavelength of light.**



John William Strutt, 3rd Baron Rayleigh
(England, 1842 - 1919)

Mie Scattering

Law of light scattering by particles equal to or grater than the **wavelength of light.**



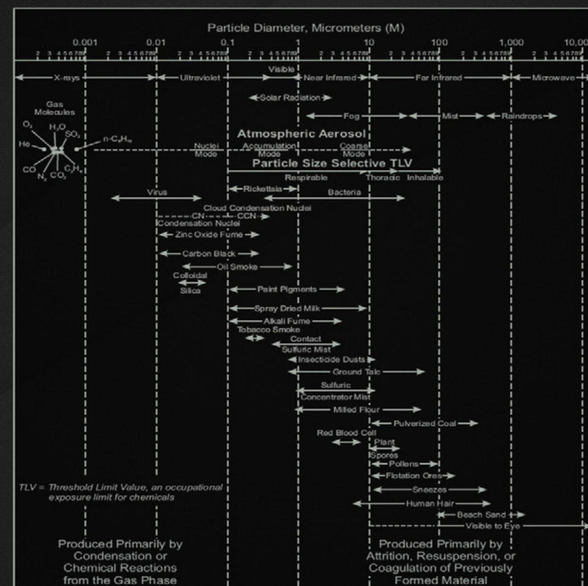
Gustav Adolf Feodor Wilhelm Ludwig Mie
(Germany, 1868 - 1957)

POLYPHONY™ DIGITAL

There are two scattering approximations in the atmospheric optics; Rayleigh scattering and Mie scattering.

Rayleigh and Mie

Particle Size Distribution of Atmosphere



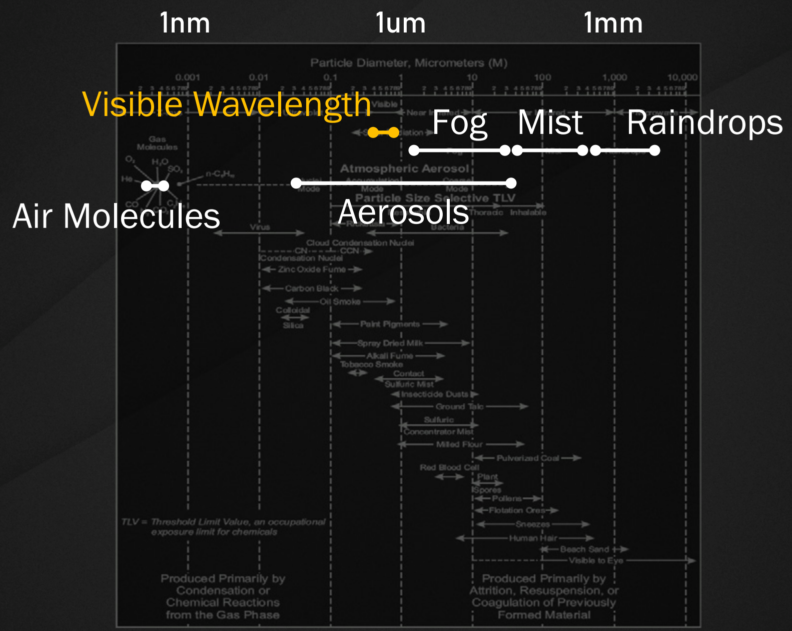
FiltrationEngineers Edge:
https://www.engineersedge.com/filtration/filtration_particle_size.htm

There are many different particles in the atmosphere.

Rayleigh scattering is a good for extremely small particles, and Mie scattering is good for rather large particles.

Rayleigh and Mie

Particle Size Distribution of Atmosphere



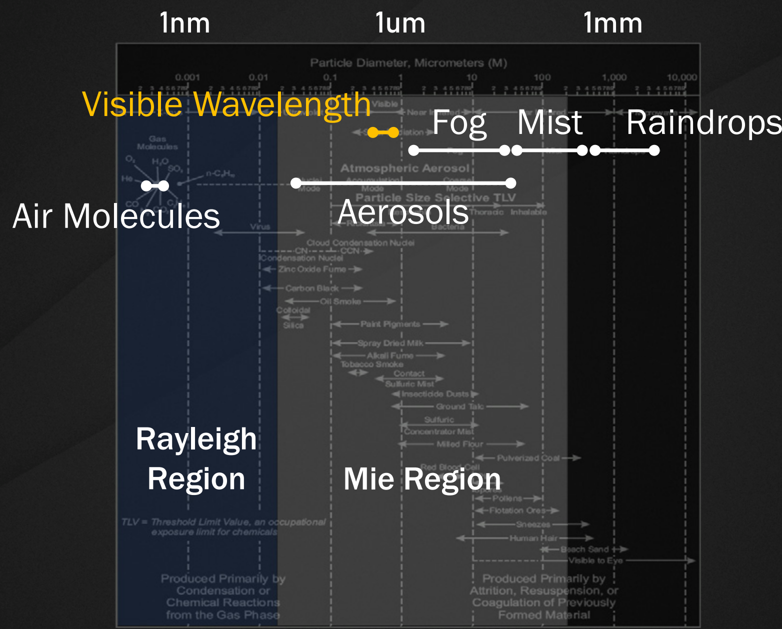
FiltrationEngineers Edge:
https://www.engineersedge.com/filtration/filtration_particle_size.htm

You may not be familiar with the term "aerosol“, it is a suspension of fine solid particles or liquid droplets in the air.

i.e.: mineral dust, city smog, sea salt, etcetera.

Rayleigh and Mie

Particle Size Distribution of Atmosphere



FiltrationEngineers Edge: https://www.engineersedge.com/filtration/filtration_particle_size.htm

Roughly speaking, they are used in this way.

You can see that Rayleigh scattering is good for air molecules and Mie scattering for aerosols and fog.

Rayleigh and Mie

Scattering Characteristics

Rayleigh Scattering

- Strongly wavelength dependent
  - Blue scatter
  - Red transmission

$$\sigma_s \propto \frac{1}{\lambda^4}$$

Mie Scattering

- Little wavelength dependent
  - White scatter

POLYPHONY™ DIGITAL

One of the differences between the two is the color of scattering.

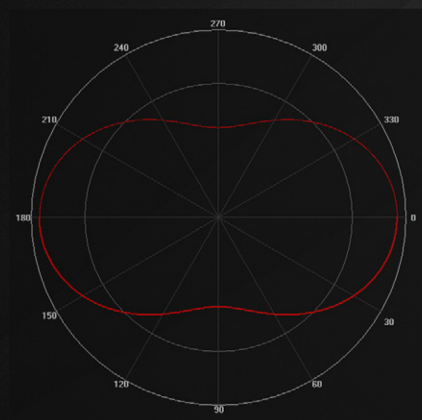
The intensity of Rayleigh scattering is inversely proportional to the fourth power of the wavelength, so shorter wavelengths such as violet and blue are scattered more.

On the other hand, the intensity of Mie scattering is mostly the same for each wavelength.
Therefore, Rayleigh scattering is blue, and Mie scattering is white.

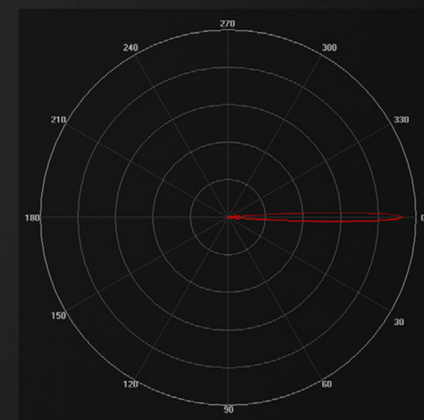
Rayleigh and Mie

Phase Functions

Rayleigh Scattering



Mie Scattering



POLYPHONY™ DIGITAL

The phase function is also different.

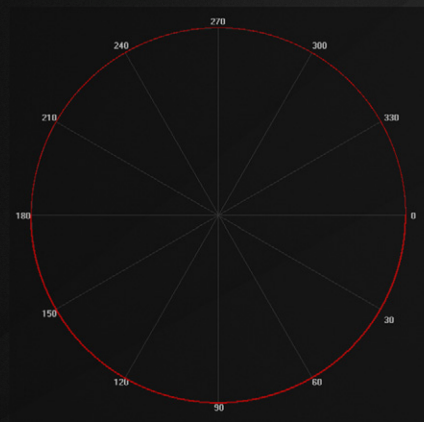
The phase function of Rayleigh scattering is symmetrical forward and backward, slightly weaker at the sides.

The phase function of Mie scattering is almost all forward.

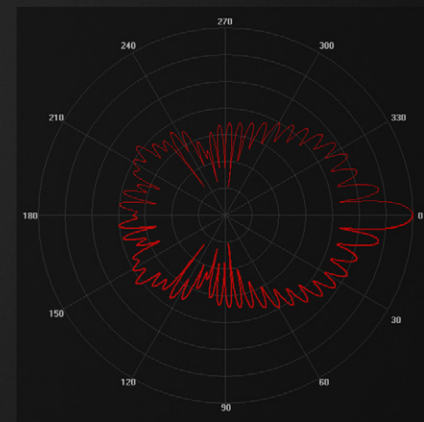
Rayleigh and Mie

Phase Functions

Rayleigh Scattering



Mie Scattering



POLYPHONY™ DIGITAL

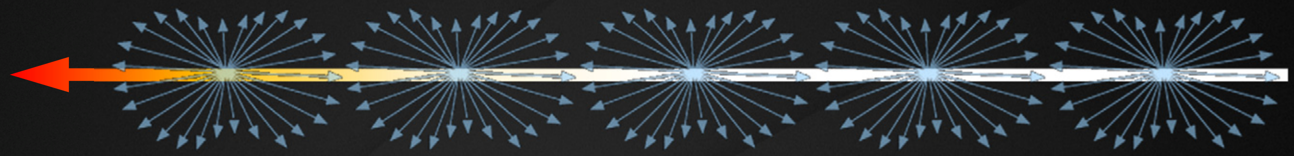
let's use a logarithmic scale. Now the plot for Rayleigh scattering has become useless but I don't care.

Because the particle size is close to the wavelength of the light, the phase function of Mie scattering has in such complex and bizarre shapes due to wave interference.

Rayleigh Scattering

The Appearance

$$\sigma_s \propto \frac{1}{\lambda^4}$$







The blue light from Rayleigh scattering makes the sky blue, and the remaining transmission becomes red which is the color of the sunset.

Mie Scattering

The Appearance







This is the Mie scattering. While clouds are typical, aerosols emit everyone's favorite God-ray.

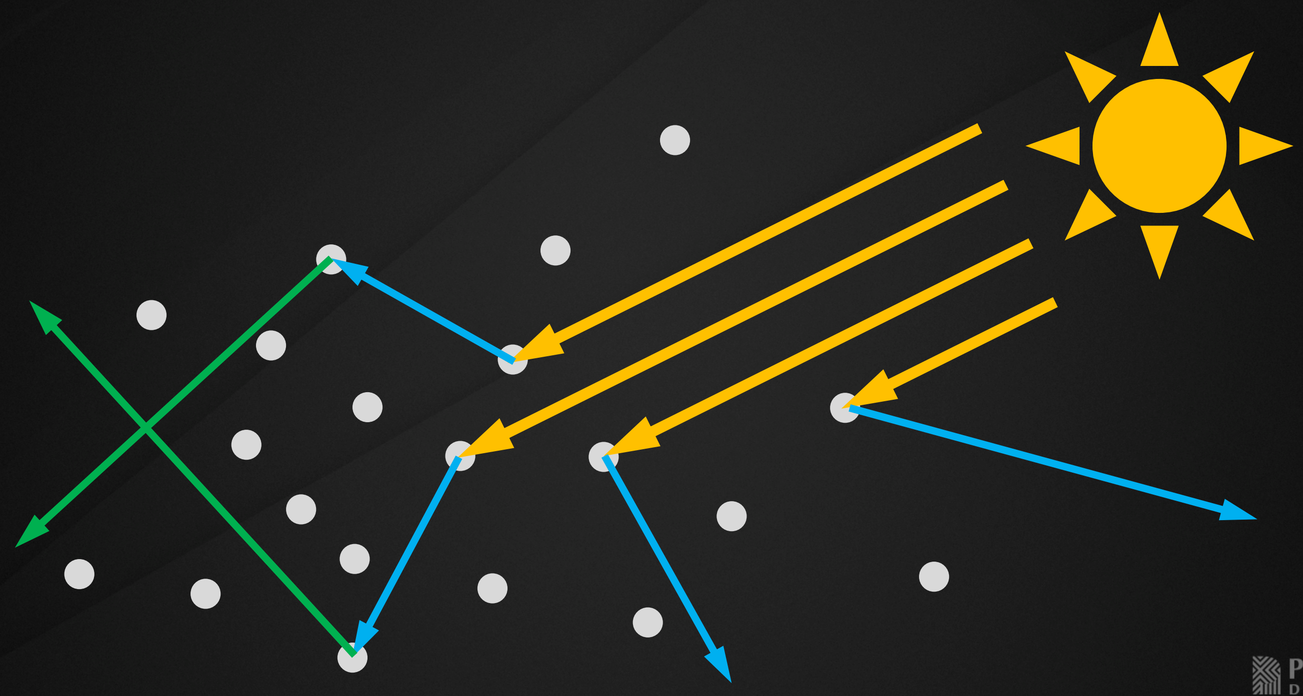
Due to the strong forward scattering, the light direction is clear.

Physics of the Sky

Multiple Scattering

Another phenomenon that cannot be ignored is multiple scattering.

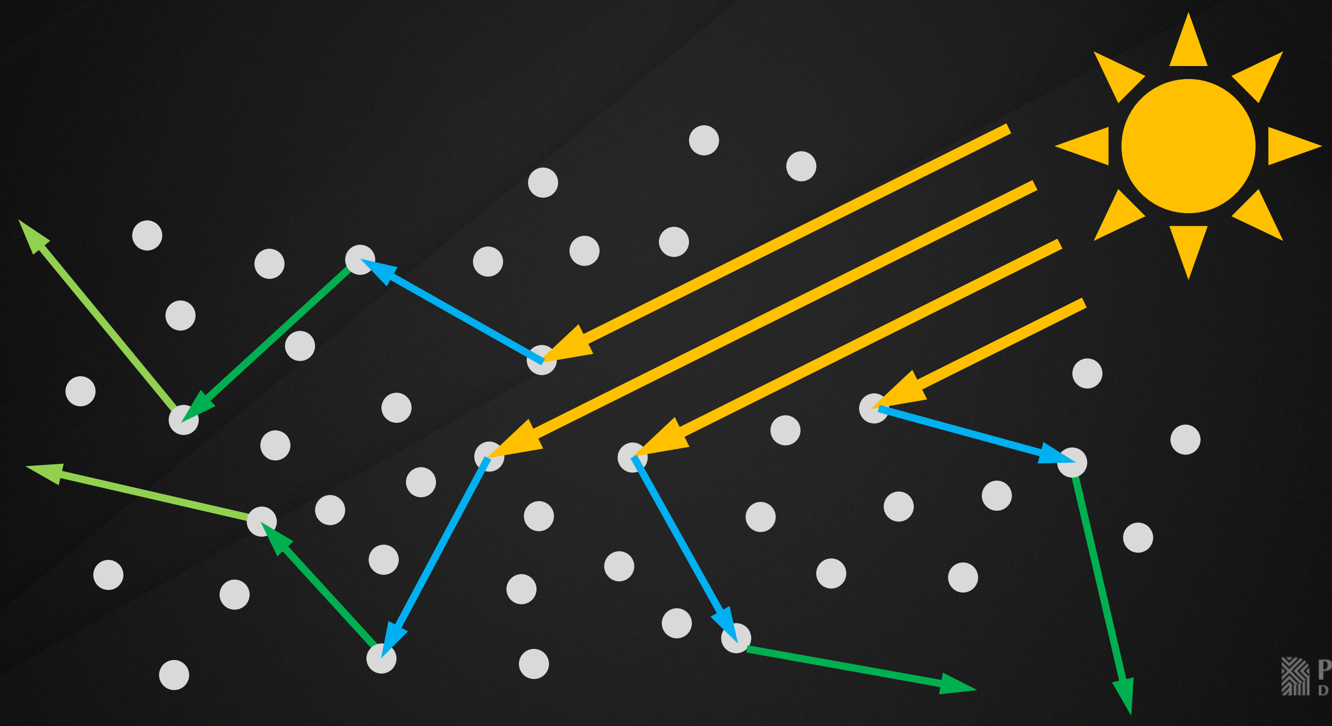
Multiple Scattering



The scattered light can be scattered again by another particle.

Yes, this is the global illumination of scattering.

Multiple Scattering



The more scattering, the more multiple scattering.

Multiple Scattering

Sunset Sky



The beautiful glow of the sunset sky is said to be produced by the multiple scattering in the long light path.

Agenda

- Introduction

- Physics of Sky
- General Discussion of Sky Simulation
- Real Earth Atmosphere
- Skysim --- our sky Renderer

- Run-time Sky Rendering Method
- Run-time Cloud Rendering Method
- Summary

POLYPHONY DIGITAL

If there is a law, it should can be simulated.

General Discussion of Sky Simulation

Let's explain the basics of sky simulation.

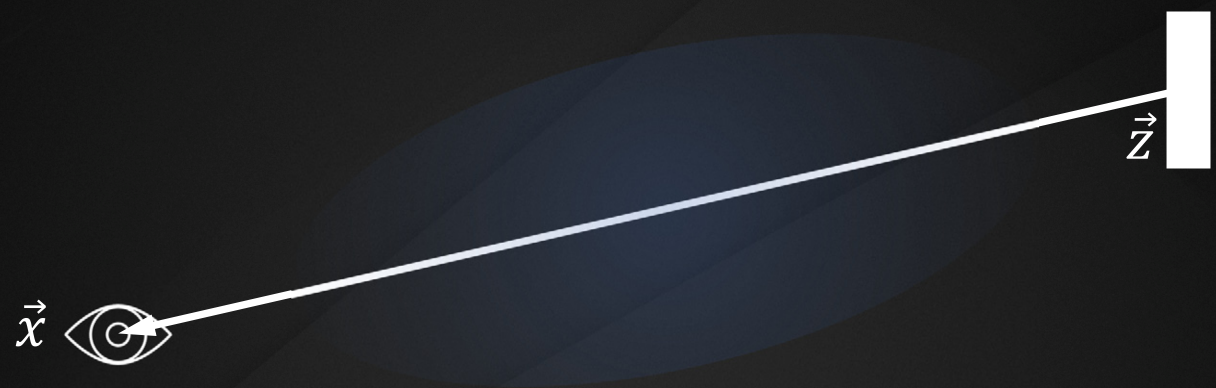
General Discussion of Sky Simulation

Volume Rendering Equation (VRE)

POLYPHONY™ DIGITAL

To physically simulate atmospheric scattering from ANY light source or viewpoint, [*] we must solve the Volume Rendering Equations.

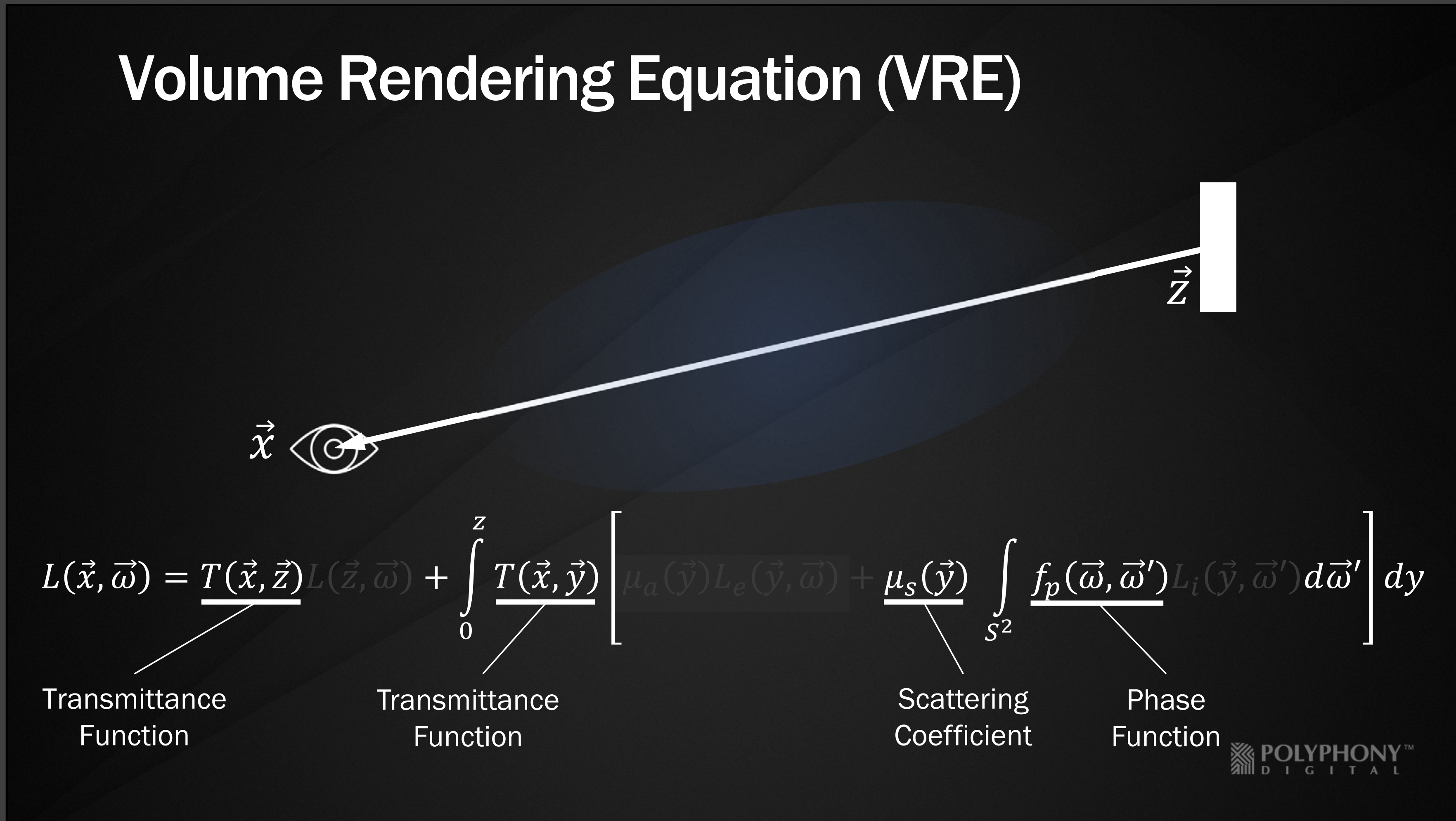
Volume Rendering Equation (VRE)



$$L(\vec{x},\vec{\omega}) = T(\vec{x},\vec{z})L(\vec{z},\vec{\omega}) + \int_0^z T(\vec{x},\vec{y})\left[\mu_a(\vec{y})L_e(\vec{y},\vec{\omega}) + \mu_s(\vec{y}) \int_{S^2} f_p(\vec{\omega},\vec{\omega}')L_i(\vec{y},\vec{\omega}')d\vec{\omega}'\right]dy$$

we must solve the Volume Rendering Equations.

It may seem quite complex but [*] only three things are related to atmosphere:



only three things are related to the atmosphere: the transmittance function, the scattering coefficient, and the phase function.

So, how do we define them?

General Discussion of Sky Simulation

Atmosphere Model in Sky Simulations

POLYPHONY™ DIGITAL

Most simulations use several approximate functions to simplify the atmosphere.

Atmosphere Models in Sky Simulations

Simplify the Atmosphere

Air (molecules)

for Rayleigh scattering

Aerosols

for Mie scattering

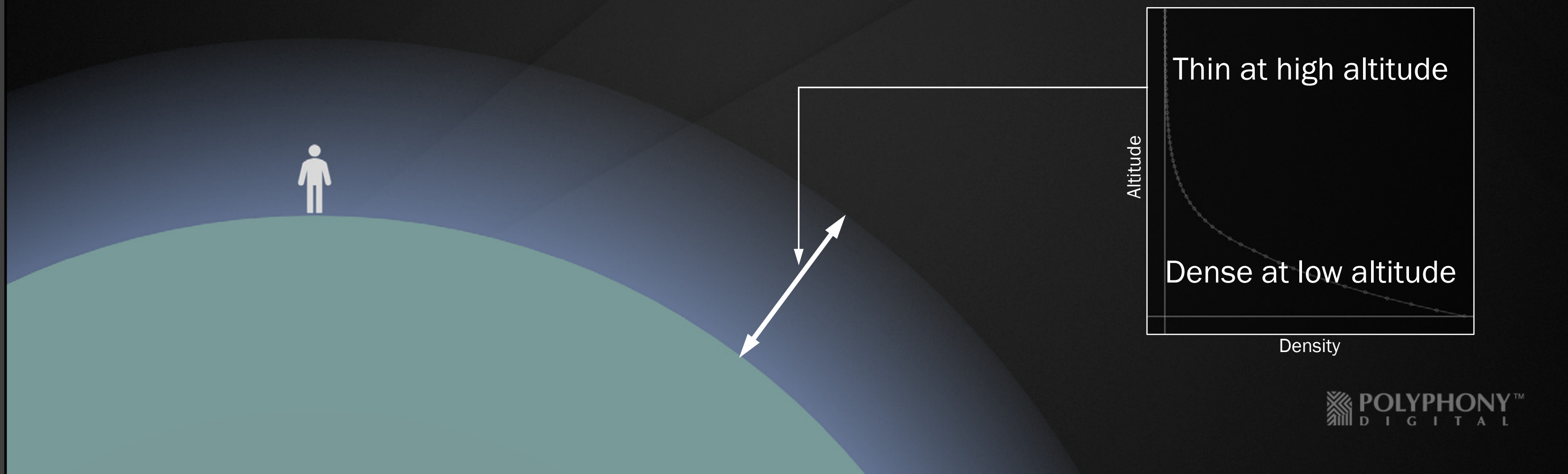
POLYPHONY™ DIGITAL

Rayleigh scattering and Mie scattering are calculated separately, so the air and the aerosol are also defined separately.

The composition of both is assumed to be constant, so that only the density varies.

Atmosphere Models in Sky Simulations

Density Profile Approximations



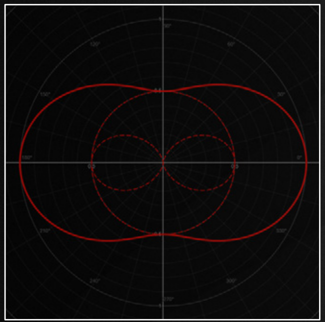
Changes in density with altitude are considered, often using an exponentially decreasing function.

Atmosphere Models in Sky Simulations

Phase Function Approximations

Rayleigh Scattering

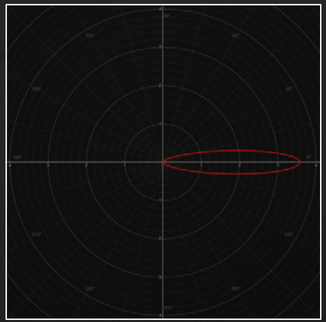
$$P_{(\theta)} = \frac{1}{4\pi} \cdot \frac{3}{4} (1 + \cos^2 \theta)$$



Mie Scattering

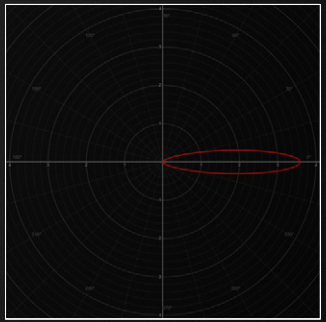
Henyey-Greenstein

$$P_{(\theta)} = \frac{1}{4\pi} \cdot \frac{1 - g^2}{(1 + g^2 - 2g\cos\theta)^{3/2}}$$



Cornette-Shanks

$$P_{(\theta)} = \frac{1}{4\pi} \cdot \frac{3}{2} \frac{(1 - g^2)}{(2 + g^2)} \cdot \frac{1 + \cos^2 \theta}{(1 + g^2 - 2g\cos\theta)^{3/2}}$$



With this assumption, the phase functions are only one each for Rayleigh scattering and Mie scattering.

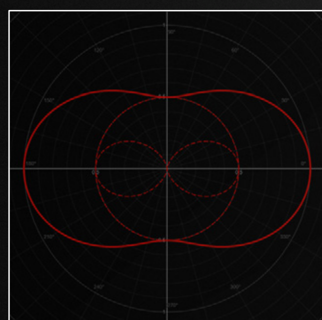
These approximate functions are often used for the phase functions.

Atmosphere Models in Sky Simulations

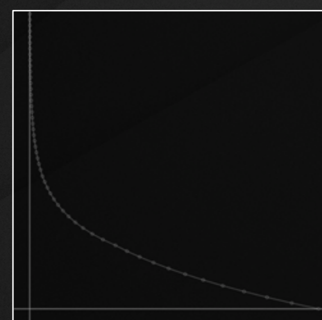
Summary

Rayleigh Scattering (for the air)

Phase Function



Density Profile

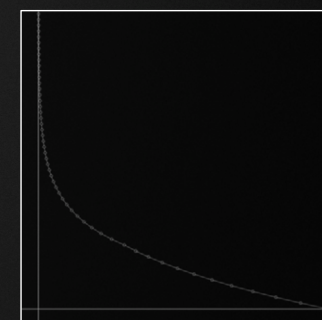


Mie Scattering (for the aerosols)

Phase Function



Density Profile

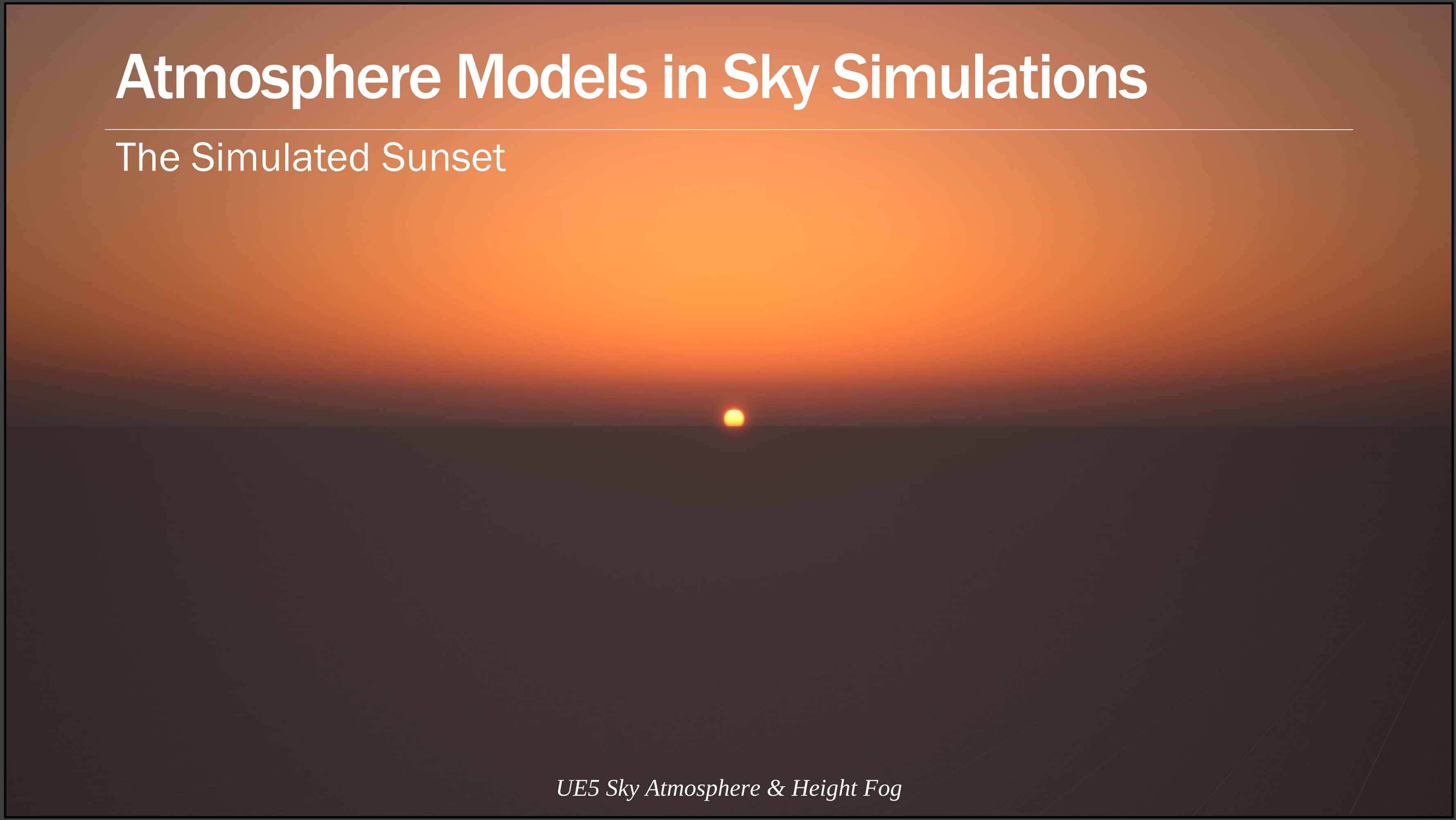


So it is common to directly define the phase function as constants and the scattering coefficient, absorption coefficient vary only the density with the altitude, for Rayleigh scattering and Mie scattering, respectively.



Photograph of Farrington Highway (Scapes)

However, the real atmosphere should be be much more complex.



UE5 Sky Atmosphere & Height Fog

We supposed the oversimplify might be the cause of the our frustration.

Atmosphere Models in Sky Simulations

Summary

- Particle Refractive Index (Distribution)
- Particle Size (Distribution)
- Particle Density (Distribution)
- Light Wavelength

▶

- Scattering Coefficient
- Absorption Coefficient
- Phase Functions

POLYPHONY™ DIGITAL

Once again, these scattering parameters are physically determined by the optical properties of the particles.

How does the approximated atmosphere different from reality?

Agenda

- Introduction

- Physics of Sky
- Simulation of Sky
- Real Earth Atmosphere (and Simulations)
- Skysim --- our sky Renderer

- Run-time Sky Rendering Method
- Run-time Cloud Rendering Method
- Summary

POLYPHONY™ DIGITAL

Now we got to the main issue.

Real Earth Atmosphere

let's verify the simulation with the reality.

Real Earth Atmosphere

Air (Rayleigh scattering)

First, about the air.

Air

Composition

- Nitrogen(N_2) : 78.08%
- Oxygen(O_2) : 20.95%
- The others : 0.97% (argon(Ar), carbon dioxide(CO_2), neon(Ne), etc…

While air contains a variety of molecules, more than 99% is just nitrogen and oxygen.

The rest seems to be negligible.

Air

Optical Properties and Size

Nitrogen (N_2)

- Refractive index: 1.00030
- Radius: 0.364nm

Oxygen (O_2)

- Refractive index: 1.00029
- Radius: 0.346nm

Let's check the refractive index and the particle size.

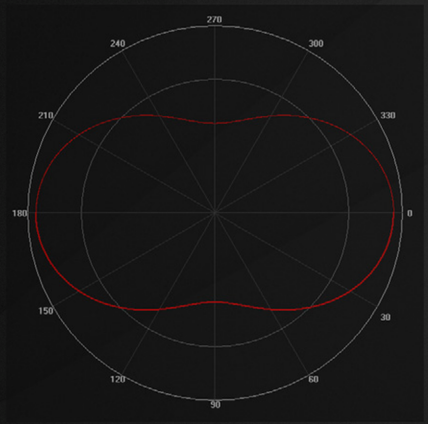
There is little difference between them.

Yes, the air composition seem to be indeed constant for the scattering.

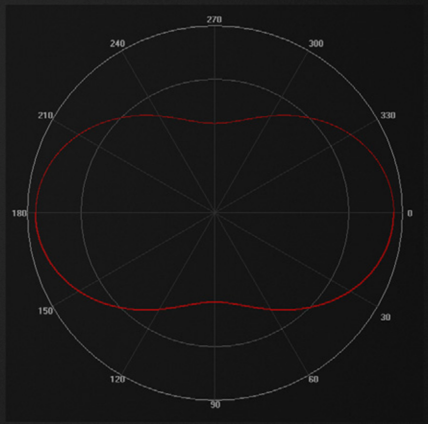
Air

Phase Functions

Nitrogen (N_2)



Oxygen (O_2)



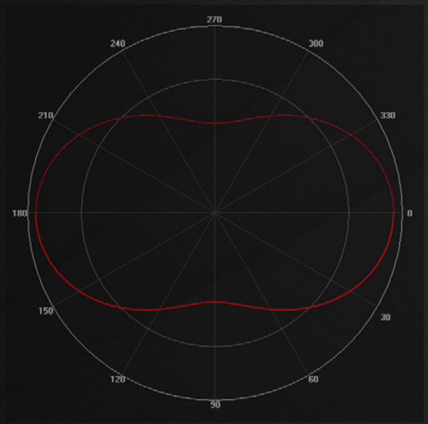
POLYPHONY™ DIGITAL

the phase function will also be almost the same.

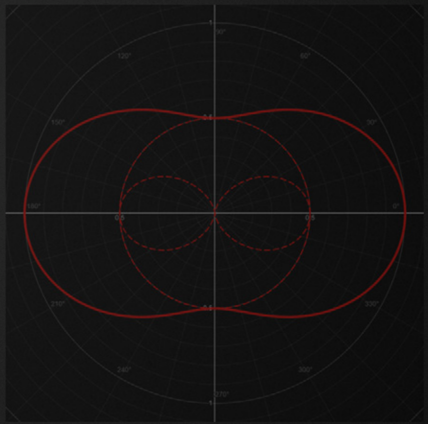
Air

Phase Function Approximations

Original Rayleigh



Approximation



POLYPHONY™ DIGITAL

Approximations is also no problem.

For extremely small particles such as air molecules, there is almost no error.

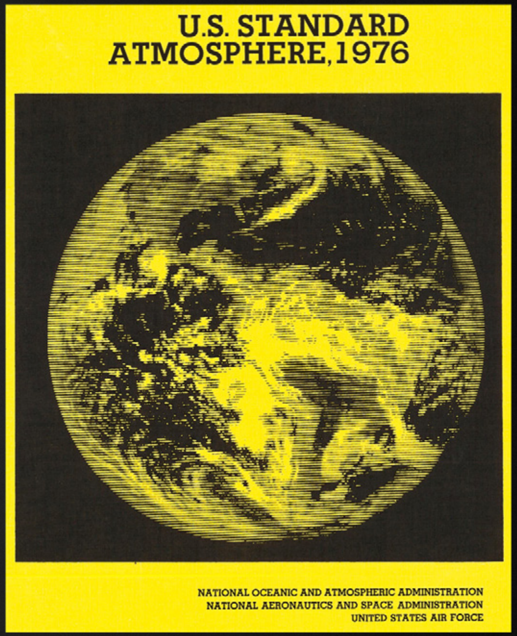
Air

Vertical Density Profile

U.S. Standard 1976 Atmosphere Model

https://www.digitaldutch.com/atmoscalc/

- Well-proven traditional model
- Up to 86km altitude
- Roughly representative of year-round, mid-latitude conditions



What about density profile?

A well-known model is the U.S. Standard.

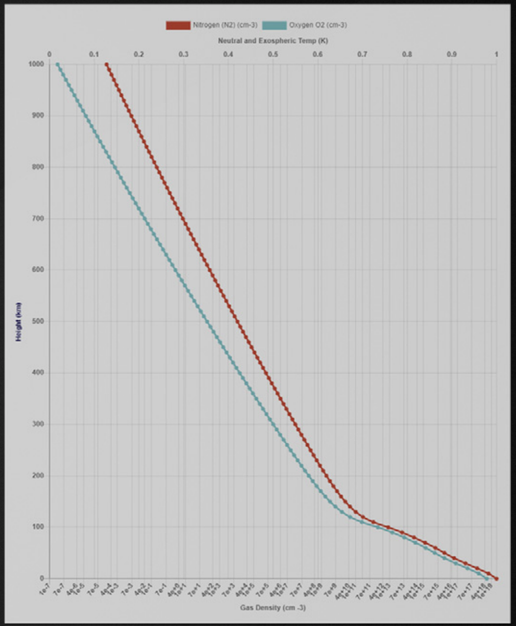
Air

Vertical Density Profile

NRLMSIS Atmosphere Model

https://kauai.ccmc.gsfc.nasa.gov/instantrun/msis

- More accurate model for astronautics and astronomy.
- Up to 1000km altitude
- Can specify:
  - Latitude
  - Longitude
  - Date and time



POLYPHONY™ DIGITAL

Here is another new model proposed for 2002; I don't know how to read it.

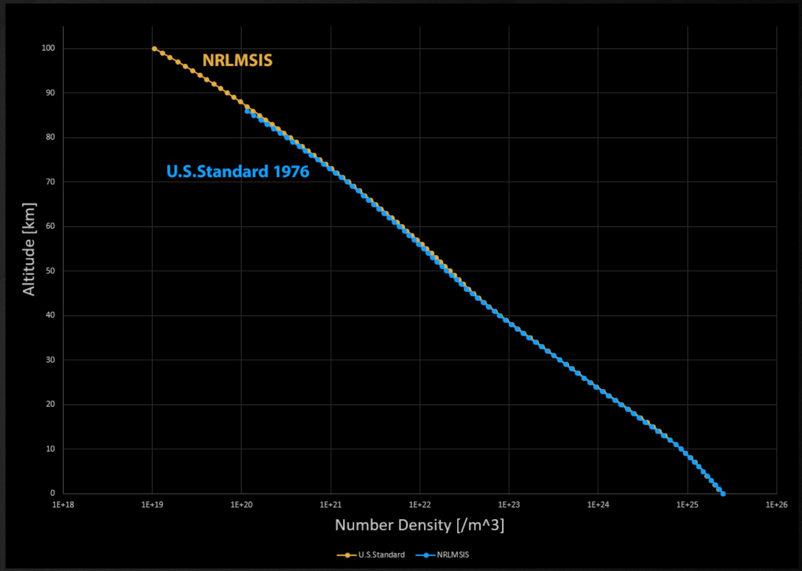
It is more advanced than U.S. Standard can specify the date and time, an observation location.

We found that the density profile did not change so much with the date and time or the latitude and longitude of the observation.

Air

Vertical Density Profile

U.S.Standard 1976 vs NRLMSIS



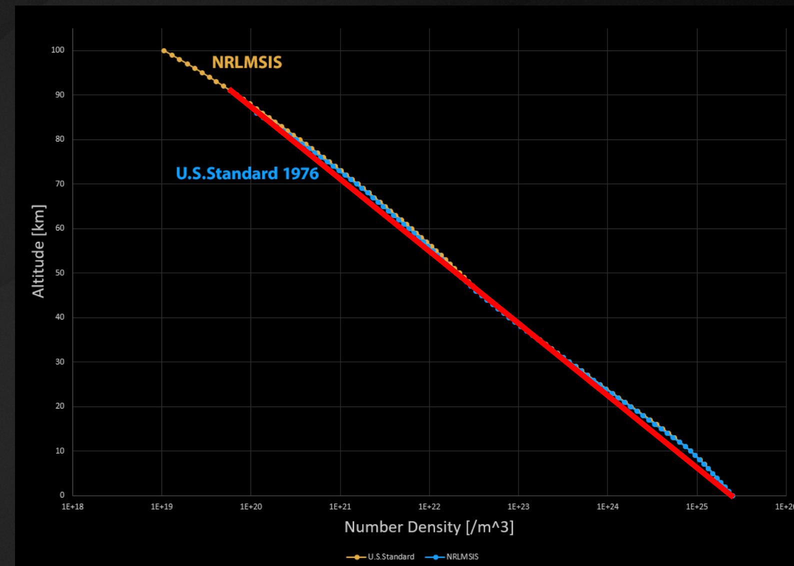
The difference between the two is extremely small, so we adopted simpler U.S. Standard.

Air

Vertical Density Profile Approximation

Exponential approximation

$$y = base_density * e^{-altitude/H}$$



POLYPHONY™ DIGITAL

As far as this logarithmic plot is concerned, it seems natural to approximate it with an exponential function.

Air

Vertical Density Profile Approximation

Exponential approximation

$$y = base_density * e^{-altitude/7.994}$$



However, it seems not so good in linear plot.

Air

Conclusion

- Nice approximation overall
- The exponential approximation for density profile is slightly doubtful



air molecules were simple and almost constant like approximation is.

We are a little concerned about the density profile.

Real Earth Atmosphere

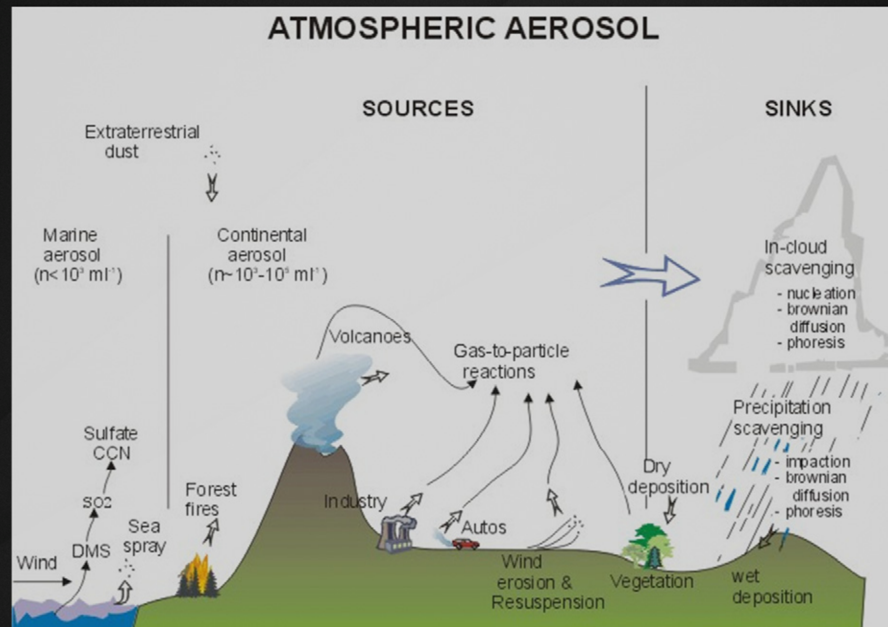
Aerosols (Mie scattering)

POLYPHONY™ DIGITAL

Next up is aerosols.

Aerosols

Composition



Complexity of Atmospheric Aerosol in Global Climate and Local Weather Image
http://butane.chem.uiuc.edu/pshapley/environmental/l12/4.html

POLYPHONY™ DIGITAL

Aerosol particles are much more complex and diverse than the air.

As shown in this figure, they have different physical properties, come from different places, and grow in different process.

Aerosols

Refractive Indices

• Water Soluble	$1.53 - 6x10^{-3}$
• Mineral Dust	$1.53 - 5.5x10^{-3}$
• Sea Salt	$1.56 - 6x10^{-3}$
• Water	$1.33 - 1x10^{-7}$
• Soot Aggregates	$1.53 - 0.43$

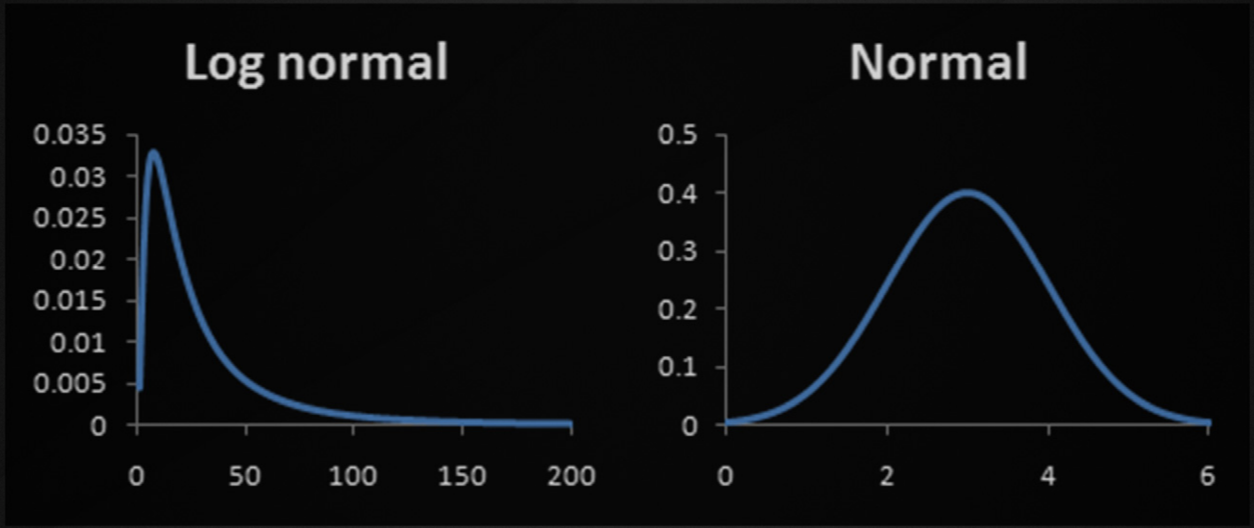
POLYPHONY™ DIGITAL

Now let's see some refractive indices of aerosols.

They are somewhat different but not much except for soot that rare aerosol.

Aerosols

Particle Size Distribution



CFI Team: https://corporatefinanceinstitute.com/resources/data-science/lognormal-distribution/

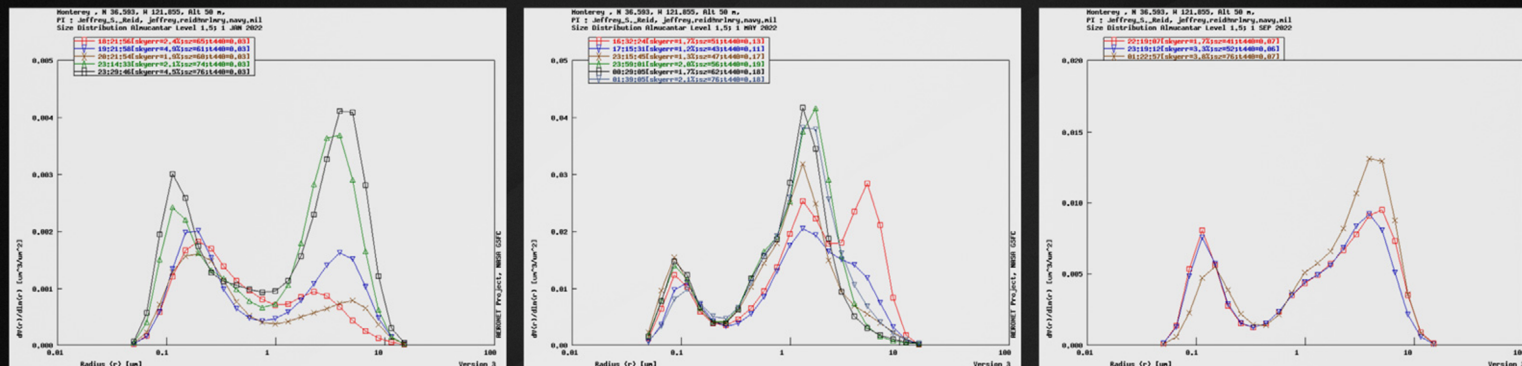
POLYPHONY™ DIGITAL

What about particle size?

The aerosols particle size distribution is known to have a log-normal distribution.

Aerosols

Particle Size Distribution



AERONET (AErosol RObotic NETwork)
https://aeronet.gsfc.nasa.gov/

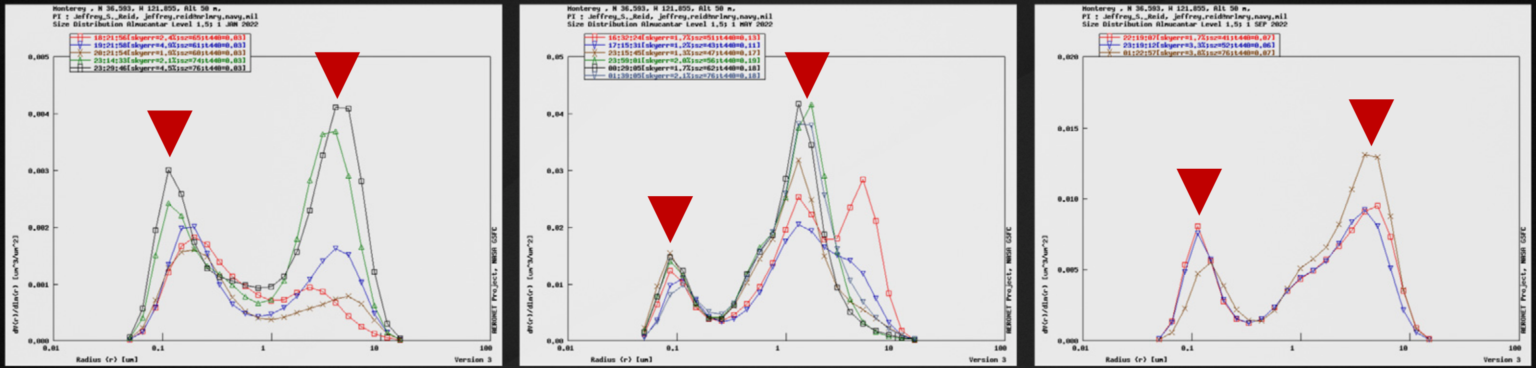
POLYPHONY™ DIGITAL

These are aerosol particle distribution measurements of AERONET.

AERONET is a global aerosol measurement network provided by NASA that was very helpful during development.

Aerosols

Particle Size Distribution

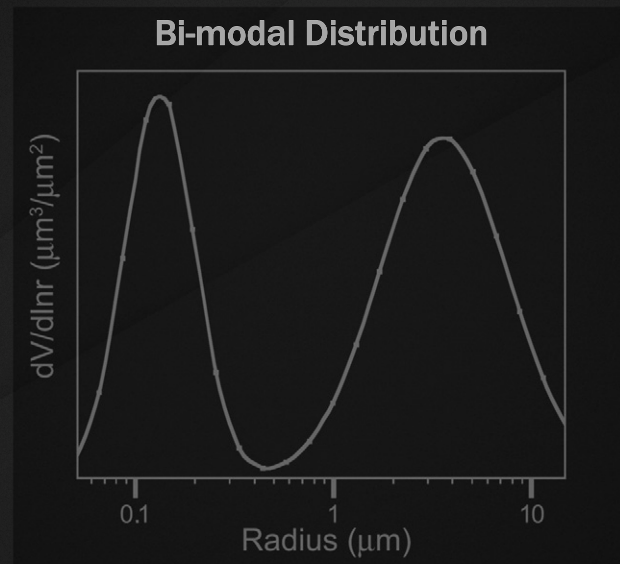


AERONET (AErosol RObotic NETwork)
https://aeronet.gsfc.nasa.gov/

You can see two peaks in any size distribution.

Aerosols

Particle Size Distribution



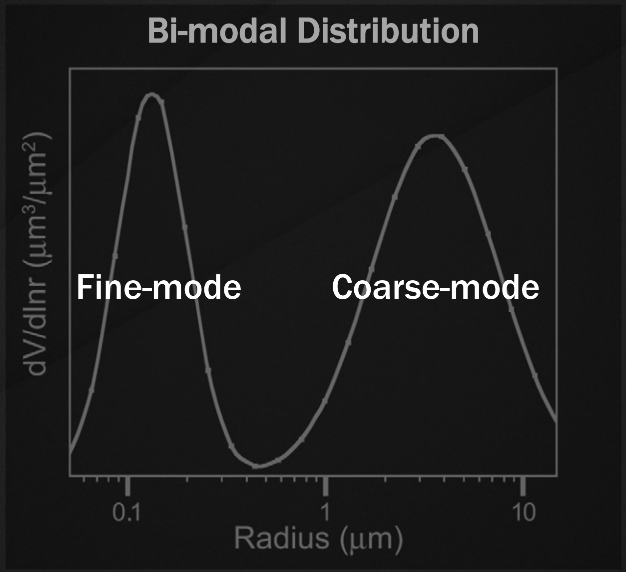
Omar et.al., 2005

POLYPHONY DIGITAL

This is called bi-modal distribution that is typical of aerosols everywhere.

Aerosols

Particle Size Distribution



Omar et.al., 2005

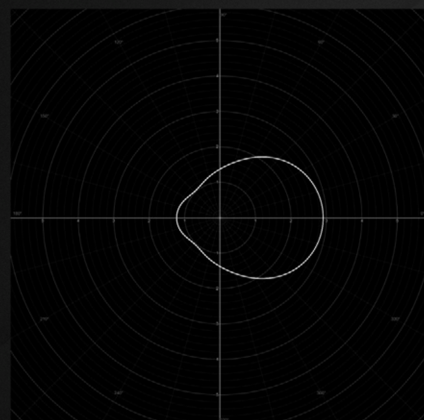
POLYPHONY DIGITAL

Smaller mode is called the fine-mode, and larger mode is called the coarse-mode.

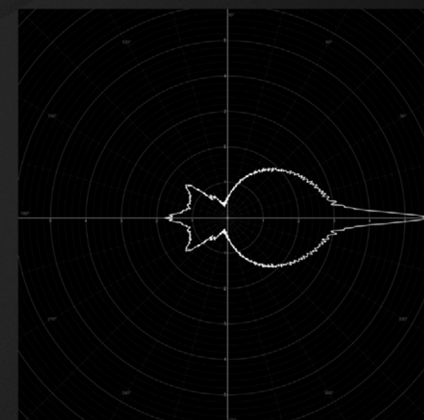
Aerosols

Phase Functions

Fine-mode



Coarse-mode



*Note that both phase functions are logarithmic scales.

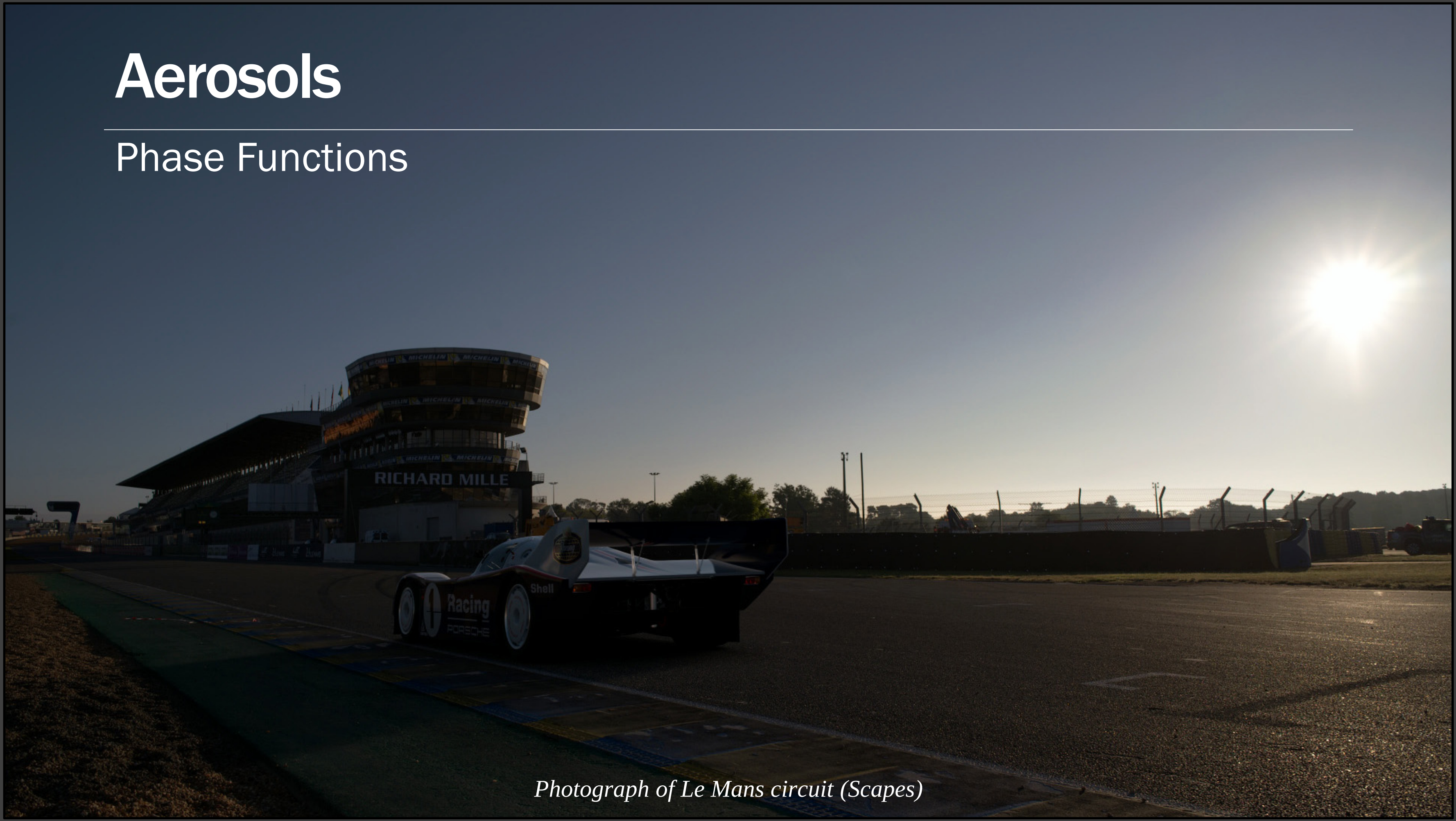
POLYPHONY™ DIGITAL

Here are the phase functions of both the fine-mode and the coarse-mode considering the log-normal distribution.

It has an even more bizarre shape than the one shown before.

Aerosols

Phase Functions



Photograph of Le Mans circuit (Scapes)

Take a look at this sky, maybe we can see the bi-modal distribution there.

Aerosols

Phase Functions

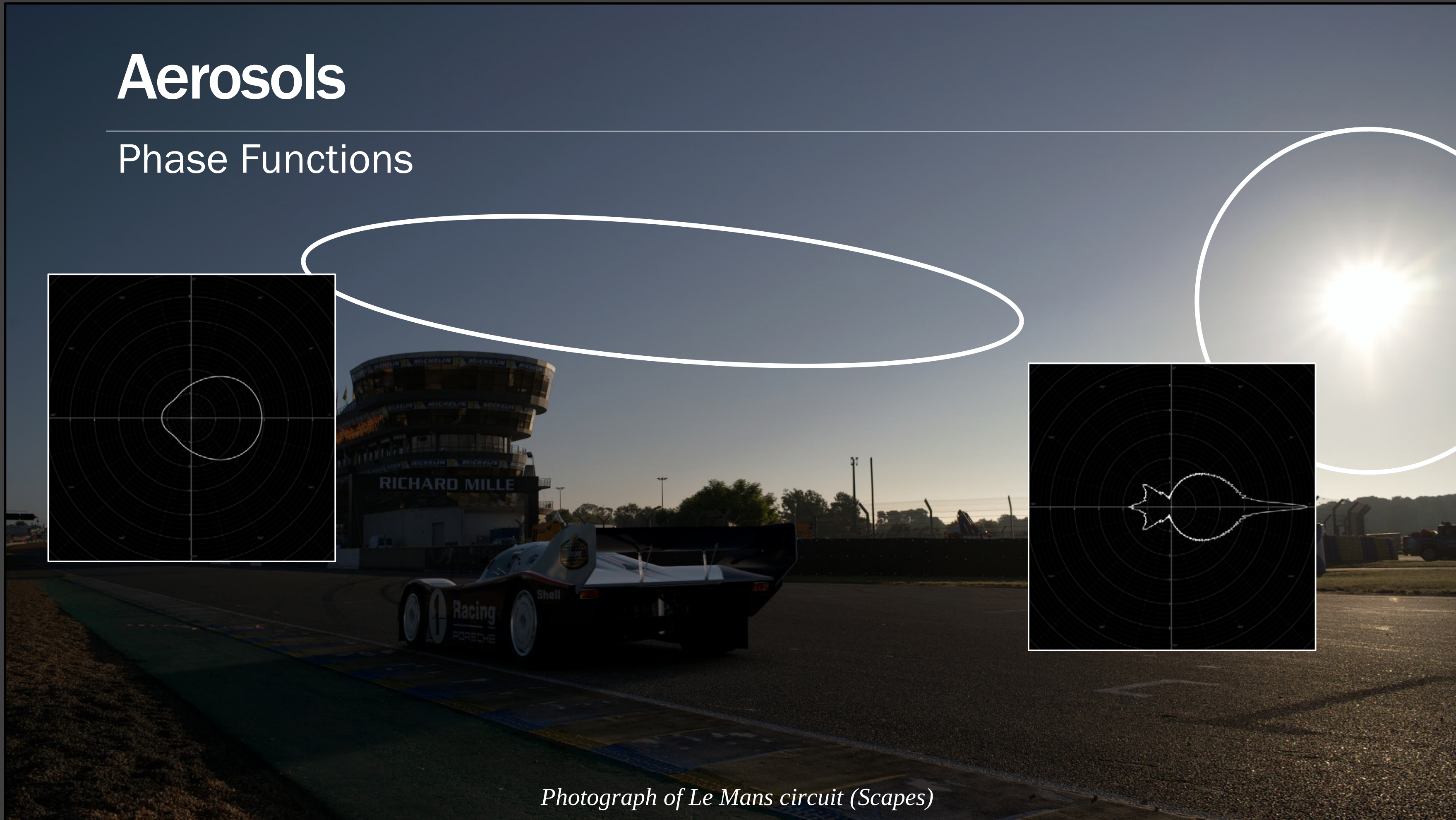


Photograph of Le Mans circuit (Scapes)

The wide, weak and dull scattering seems to be from fine-mode aerosols.

Aerosols

Phase Functions



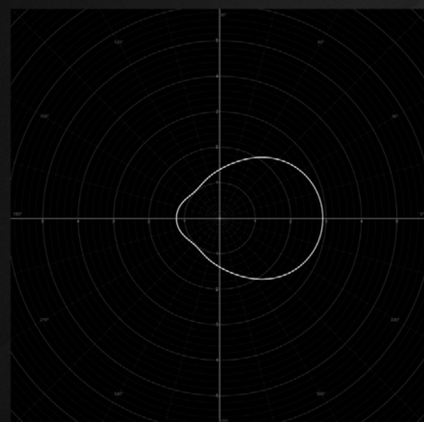
Photograph of Le Mans circuit (Scapes)

This strong and sharp scattering near the sun seems to be from coarse-mode aerosols.

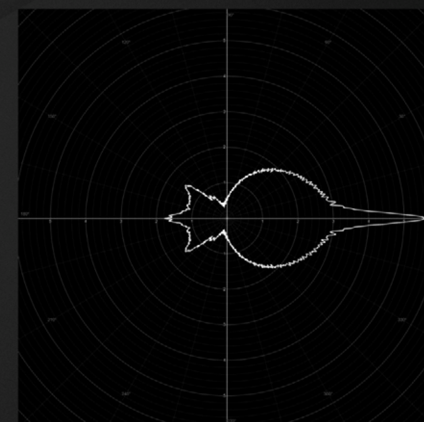
Aerosols

Phase Function Approximation

Fine-mode



Coarse-mode



*Note that both phase functions are logarithmic scales.

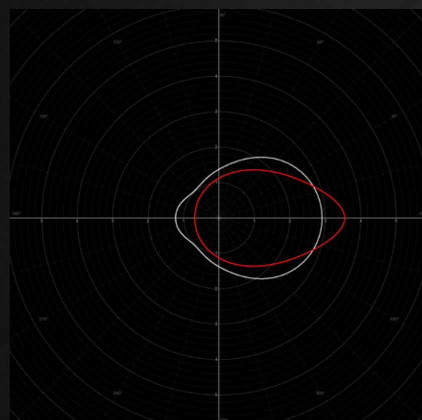
POLYPHONY™ DIGITAL

What about the accuracy of the approximation functions?

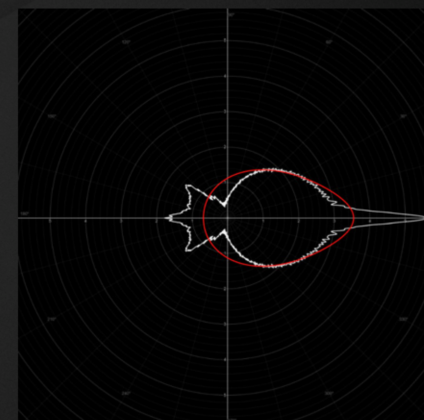
Aerosols

Phase Function Approximation

Fine-mode



Coarse-mode



*Note that both phase functions are logarithmic scales.

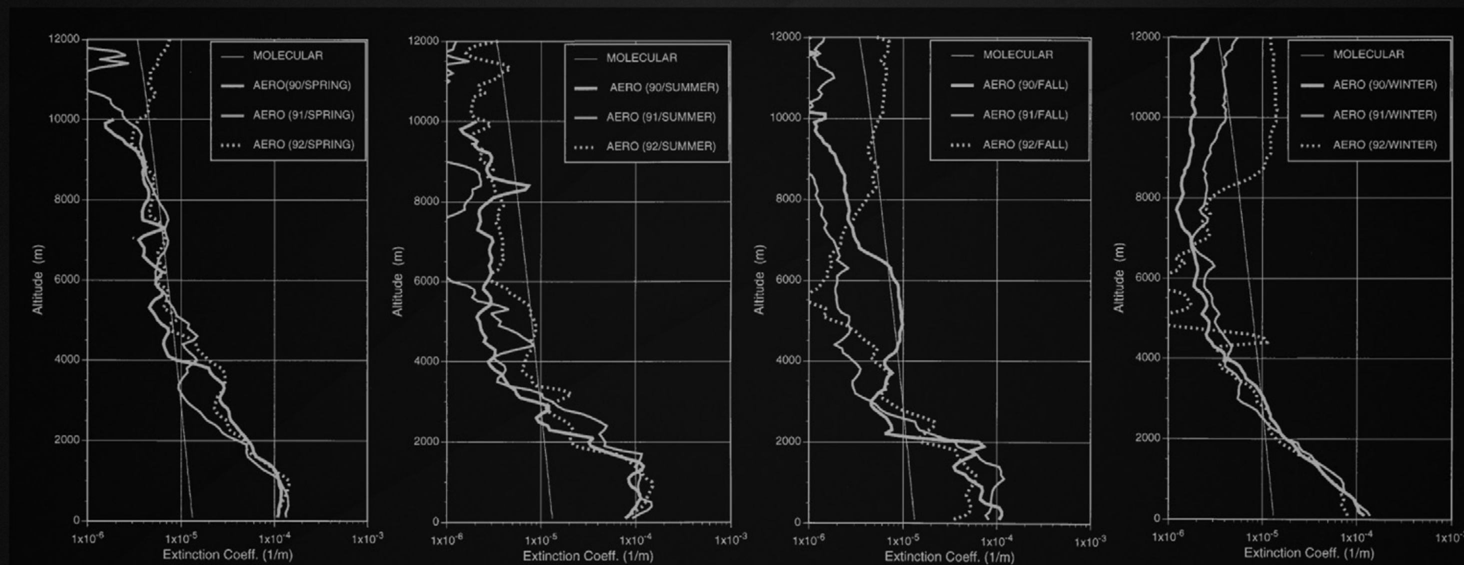
POLYPHONY™ DIGITAL

We tried to approximate with the Henyey-Greenstein phase function which is the red line.
but it can never fit well enough.

Since they are plotted on the logarithmic scales, note that the error would be more than 16 times in the coarse-mode forward scattering.

Aerosols

Vertical Density Profile



Sasano et al., 1996

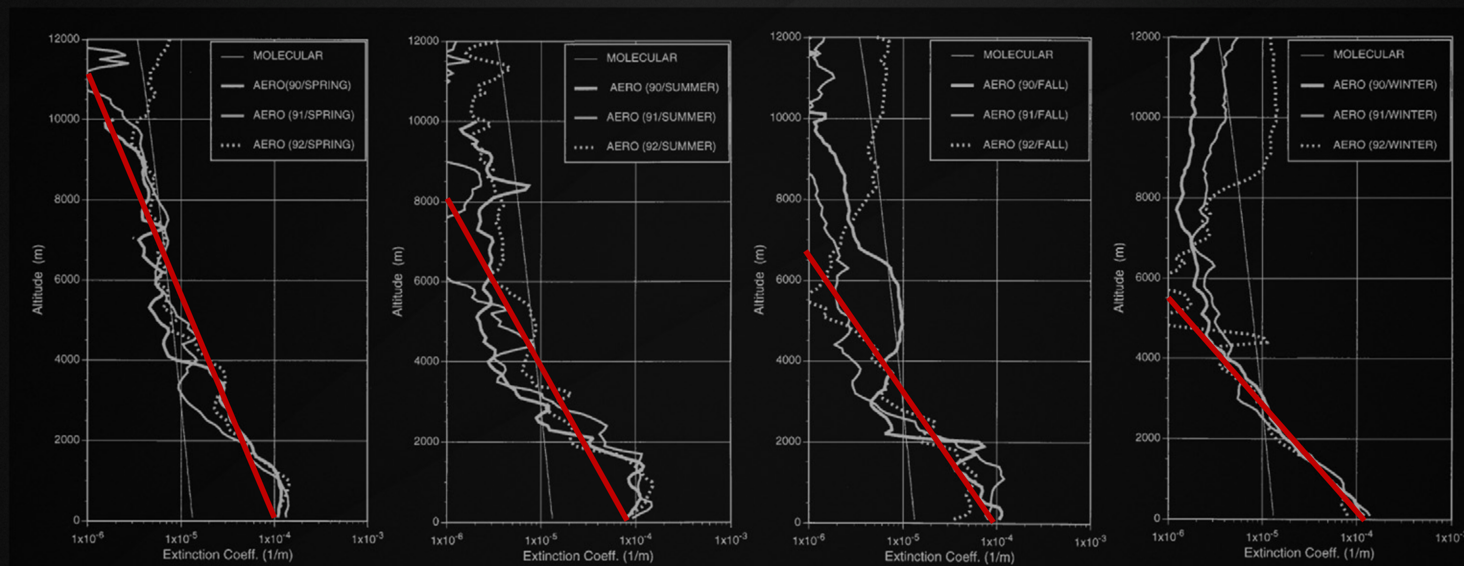
POLYPHONY DIGITAL

Okay, What about the density profile?

Even at first glance, it seems so complex and far from a simple function.

Aerosols

Vertical Density Profile Approximation



Sasano et al., 1996

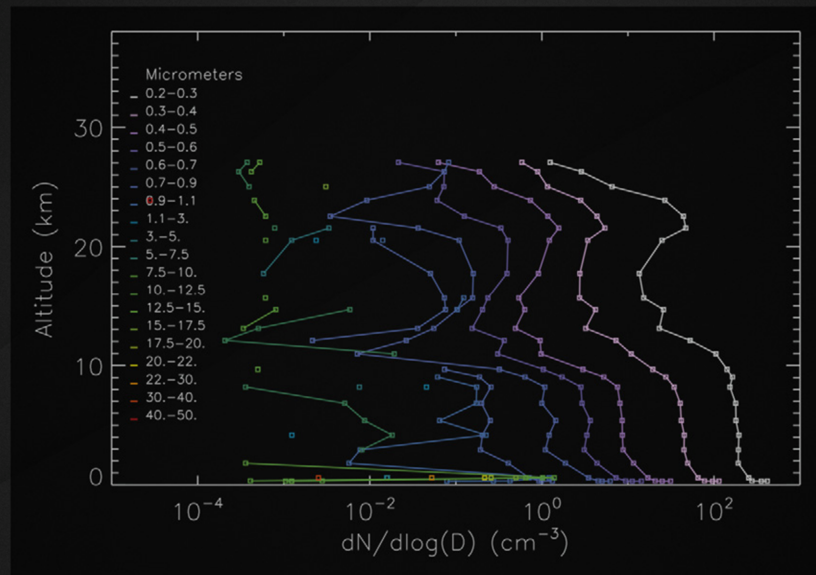
Red line: exponential approximation

POLYPHONY™ DIGITAL

the exponential approximation, which is red line, seems to have too large error.

Aerosols

Vertical Density Profile



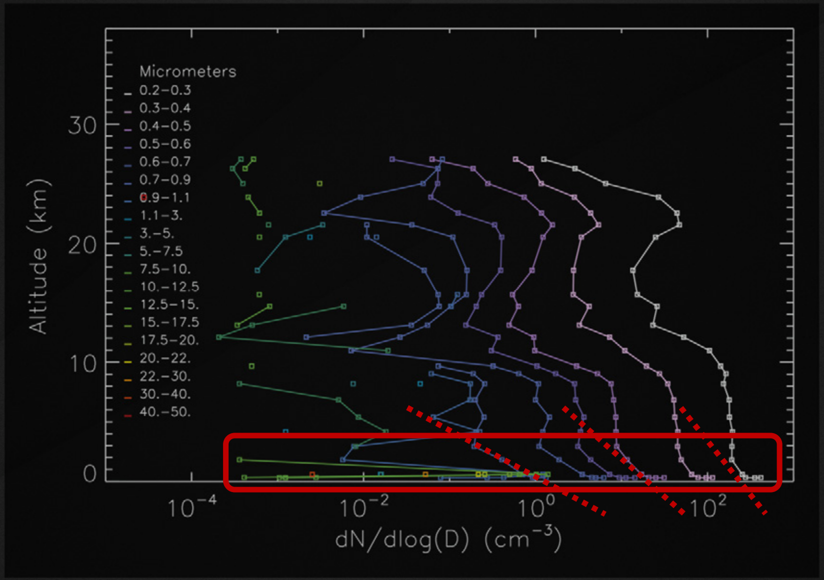
Brattich et al., 2019

POLYPHONY™ DIGITAL

This invaluable figure shows the density profile for each particle size bins.

Aerosols

Vertical Density Profile



Brattich et al., 2019

It shows that the density profiles also vary with particle size.

Aerosols

Conclusion

- Errors of Phase Function:
  - size distribution ignored
  - Lack of forward scattering sharpness

- Errors of Vertical Density Profile



POLYPHONY DIGITAL

Many differences were found, or perhaps I should say, nothing fits enough.

Real Earth Atmosphere

Between Rayleigh and Mie

In addition, there is an interesting thing I would like to tell you.

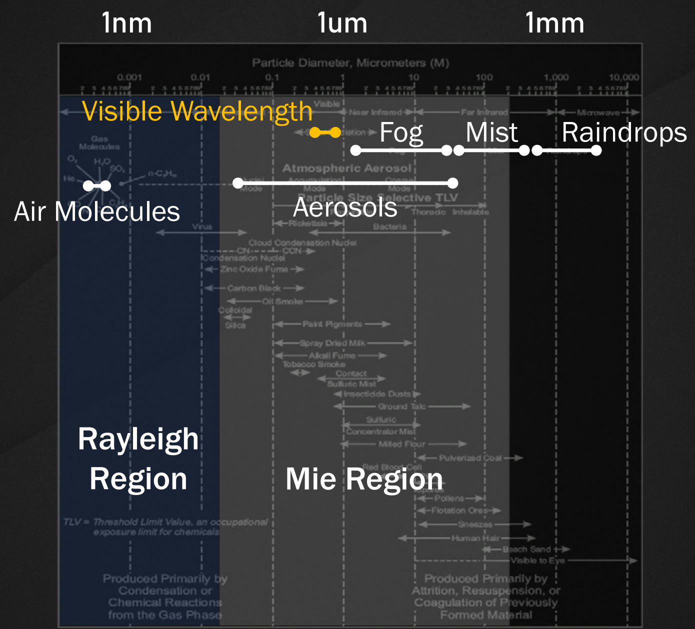
‘Mie scattering has no wavelength dependence.’

POLYPHONY™ DIGITAL

It is very often said that Mie scattering has no wavelength dependence in the field of computer graphics.

However, is it true?

Between Rayleigh and Mie

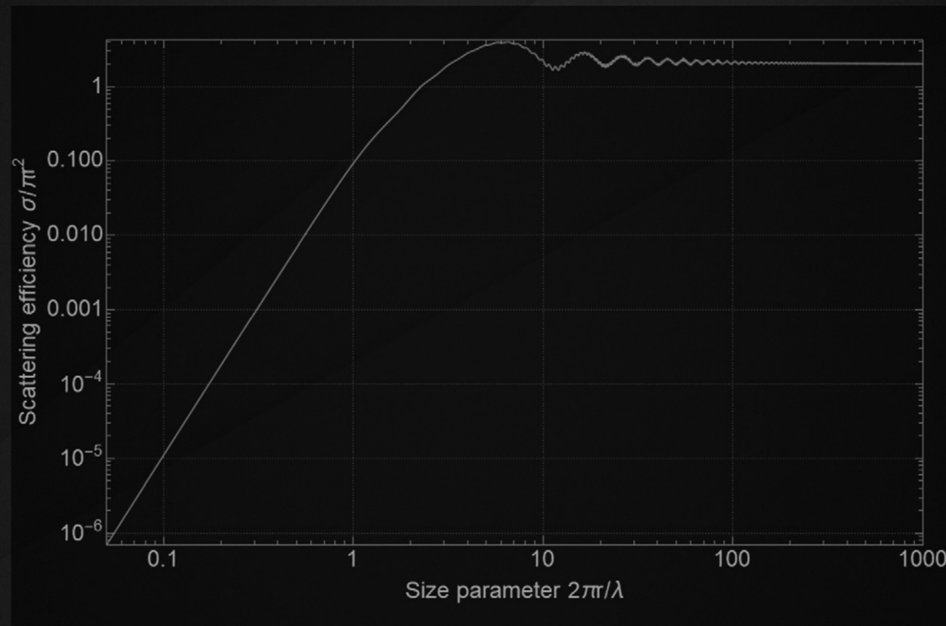


FiltrationEngineers Edge:
https://www.engineersedge.com/filtration/filtration_particle_size.htm

It’s NOT true.

Rayleigh scattering and Mie scattering are not binary, they are both only conditional approximations that are created separately.

Between Rayleigh and Mie

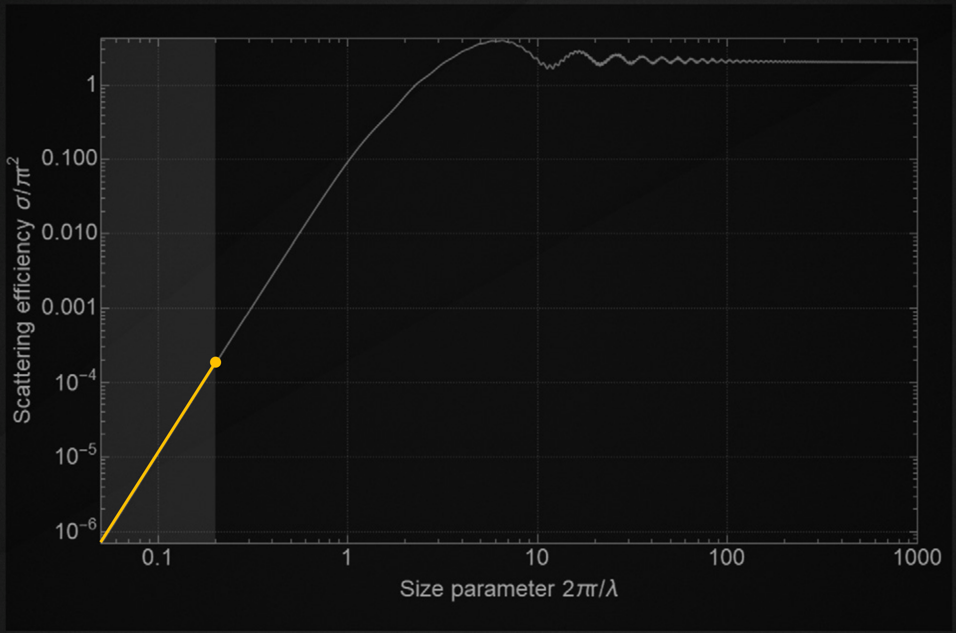


POLYPHONY™ DIGITAL

This is a plot of the scattering efficiency versus size parameter.

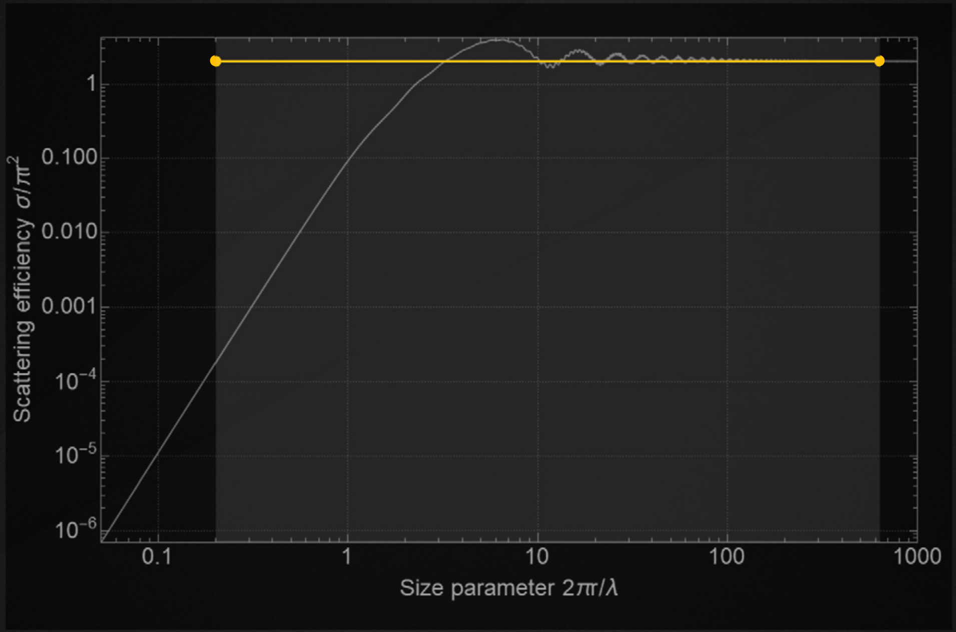
Think of the slope of this plot represents the strength of the wavelength dependence.

Between Rayleigh and Mie



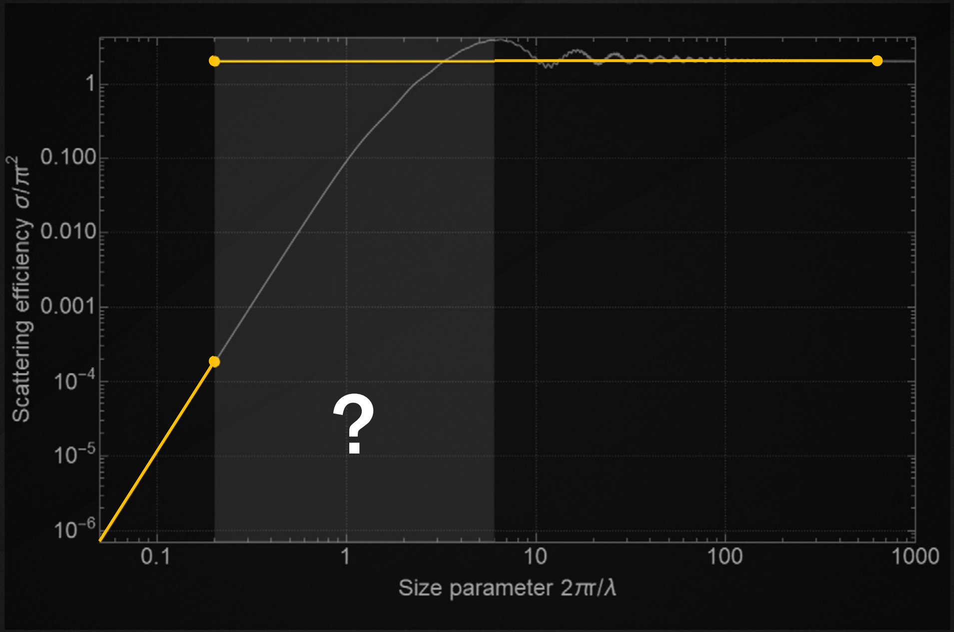
In general, here is the Rayleigh scattering region.

Between Rayleigh and Mie



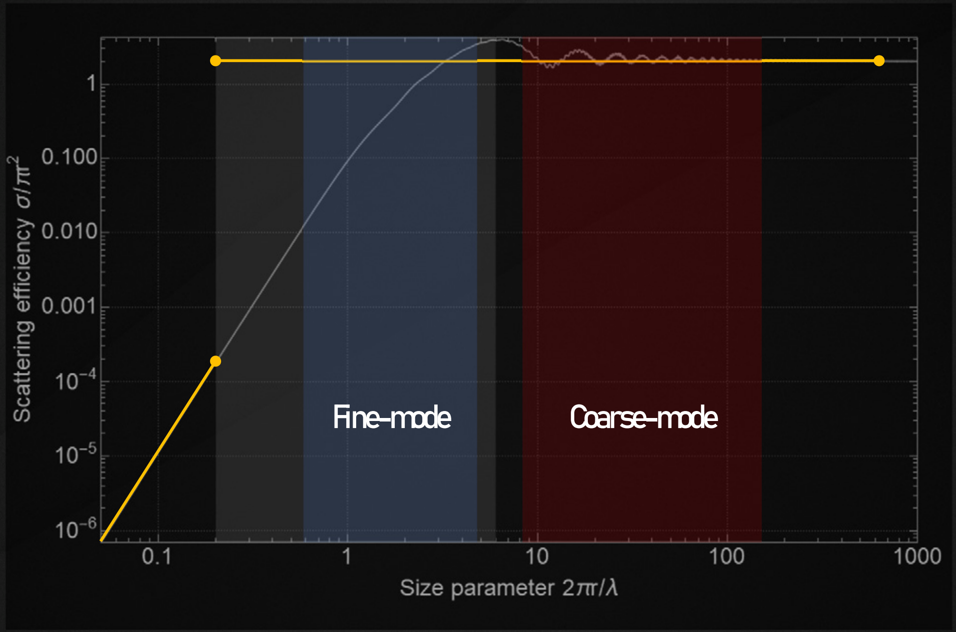
And here is the Mie scattering region.

Between Rayleigh and Mie



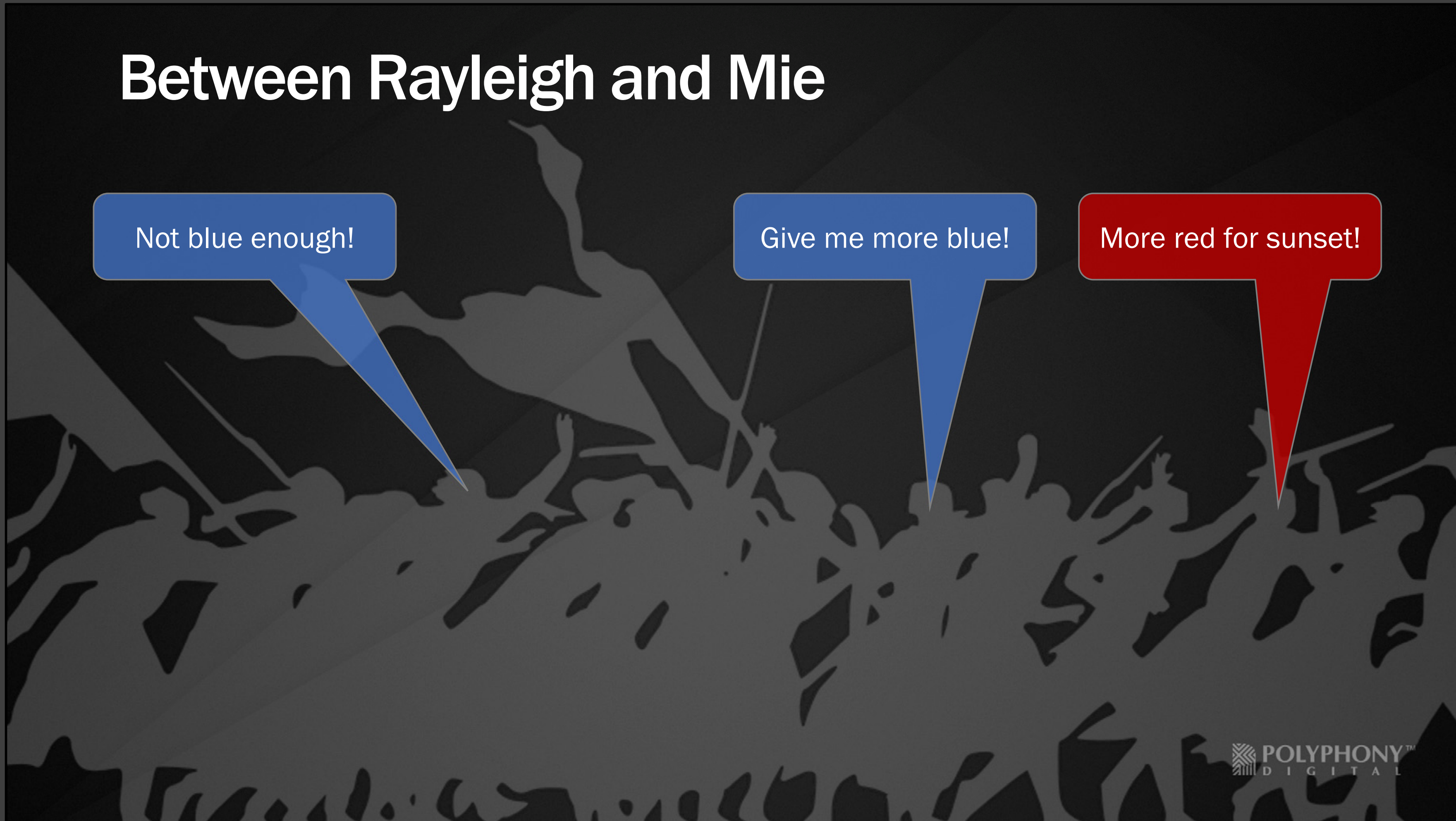
However, there is a region that just cannot be approximated exactly by Rayleigh scattering but is still very close.

Between Rayleigh and Mie



The size parameters of fine-mode aerosols are exactly in this region.

Yes, fine-mode Mie scattering is actually quite blue and could make the sunset red like the Rayleigh scattering.



I think this is one of the reasons why artists always complain about the lack of blue in the fog and red in the sunset.

Note that although the color is similar to Rayleigh scattering, the phase function is still that of Mie scattering.

Agenda

- Introduction

- Physics of Sky
- Simulation of Sky
- Real Earth Atmosphere (and Simulations)
- Skysim --- Our Sky Renderer

- Run-time Sky Rendering Method
- Run-time Cloud Rendering Method
- Summary

Was it interesting? or did you already know about it?

Okay, our survey is finally finished.

Skysim

Our Sky Renderer

Let’s take the next step.

Skysim

Our Sky Renderer

Difference between real atmospheric scattering and existing simulations

- Phase functions of Mie scattering
- Wavelength dependence of Mie scattering
- Vertical density profiles of both air molecules and aerosols

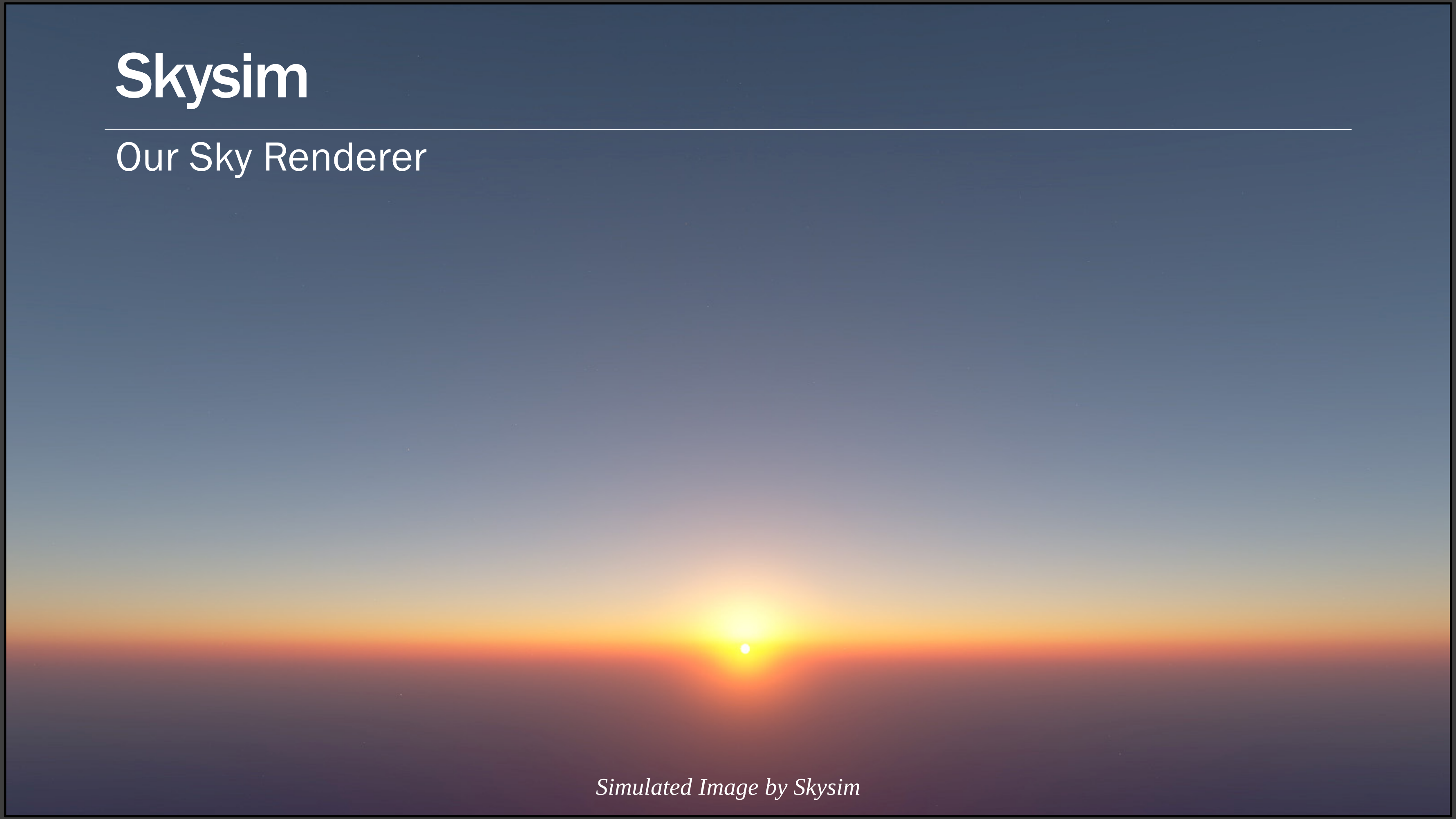
Some of the difference between the simulations and the reality have been revealed.

However, it is no surprise that existing simulations are not perfect.

The question is how far can we improve them?

Skysim

Our Sky Renderer



Simulated Image by Skysim

To make sure that, we decided to develop Renderer ignoring computation costs.

Skysim

Our Sky Renderer

Existing simulations		Skysim
Constant	Composition	Various, Measurements-based
No	Size distribution	Measurements-based bi-modal distribution
approximate functions	Phase function	Derived from optical properties by BHMIE
Exponential functions	Density profiles	Measurements-based table

POLYPHONY DIGITAL

Because the errors of the Mie scattering seem to be so large, we found it necessary to physically derive the scattering coefficient, absorption coefficient, and phase function instead of defining them directly.

Skysim

Requirements

- The scattering parameters are derived from physical properties of the particles at each coordinate.
- Atmosphere model format that can use measured data as is with no loss
- Light transport simulation as accurate as atmosphere model

The main issue is to simulate physically accurate scattering based on highly accurate atmosphere model.

In addition, we got a lot of measurement through the survey and we want to use it as is.

Skysim

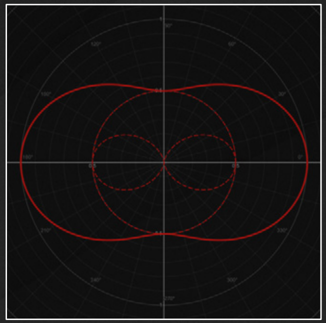
Modeling of the Air and the Aerosols

Now let me tell you exactly how we modeled the atmosphere.

Modeling of the Air

- There is no problem with the Rayleigh scattering phase functions.

$$P_{(\theta)} = \frac{1}{4\pi} \cdot \frac{3}{4}(1 + \cos^2\theta)$$



- For the vertical density profile, we use the U.S.Standard 1976

For air molecules, there is no major innovations have been made except using U.S.Standard as vertical density profile.

Modeling of the Air

Sky with Only Rayleigh Scattering



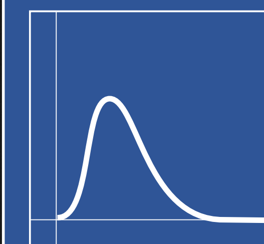
Yeah, That’s it.

Modeling of the Aerosols

The Structure of Aerosol Layer

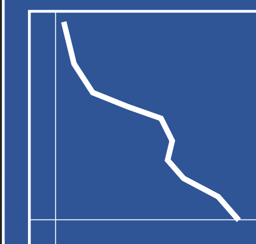
Physical properties

- Refractive index
- Size distribution



Vertical density profiles

- Aerosol ID
- Density table



POLYPHONY™ DIGITAL

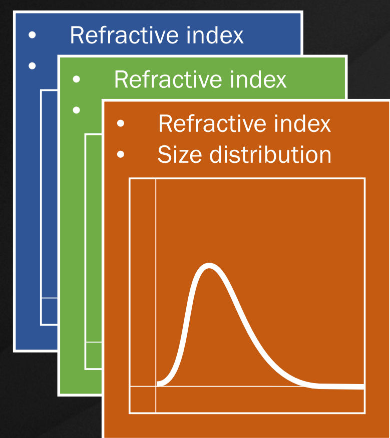
Here is the key, aerosols.

Aerosol layers are defined by pair of the physical properties of the particles and the vertical density profile table.

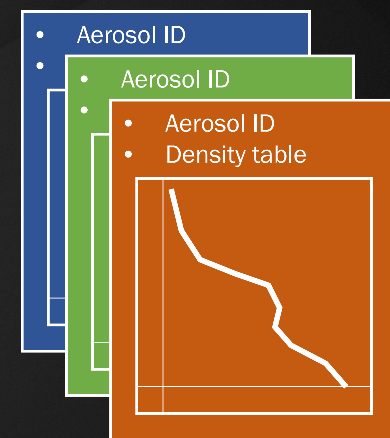
Modeling of the Aerosols

The Structure of Aerosol Layer

Physical properties



Vertical density profiles



POLYPHONY™ DIGITAL

The layer can be added as needed.

Modeling of the Aerosols

How to Define the Optical Properties of Particles

Complex refractive index

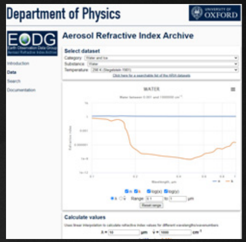
- Real part
- Imaginary part

Log-normal size distribution

- Mean radius
- Sigma (GSD)

Vertical density profile







RefractiveIndex.INFO
https://refractiveindex.info/

Aerosol Refractive Index Archive –EODG
http://eodg.atm.ox.ac.uk/ARIA/data

The aerosol database from OPAC –GEISA
https://geisa.aeris-data.fr/opac/

POLYPHONY DIGITAL

The wavelength dependent refractive index can be easily obtained from these sites.

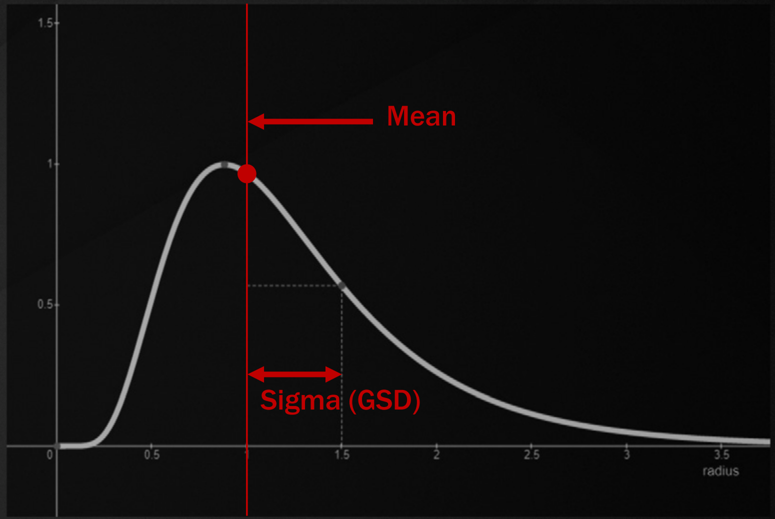
Modeling of the Aerosols

How to Define the Size Distributions

Complex refractive index
- Real part
- Imaginary part

Log-normal size distribution
- Mean radius
- Sigma (GSD)

Vertical density profile



POLYPHONY™ DIGITAL

The log-normal distribution can be defined by the mean radius and the geometric standard deviation; You can think of it as the standard deviation in a lognormal distribution.

Modeling of the Aerosols

How to Define the Density Profiles

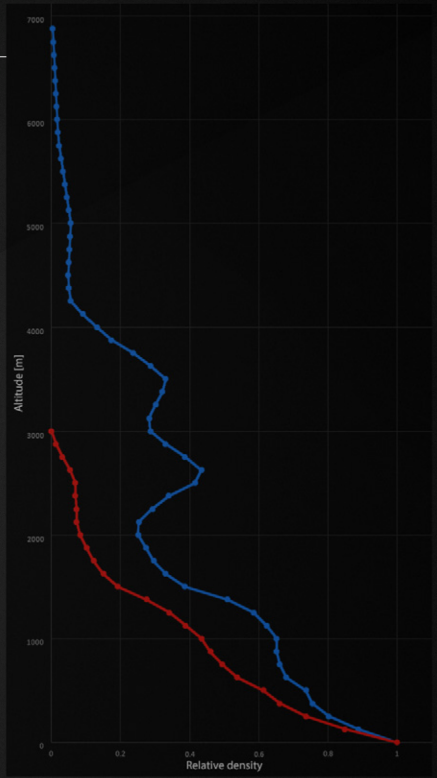
Complex refractive index

- Real part
- Imaginary part

Log-normal size distribution

- Mean radius
- Sigma (GSD)

Vertical density profile

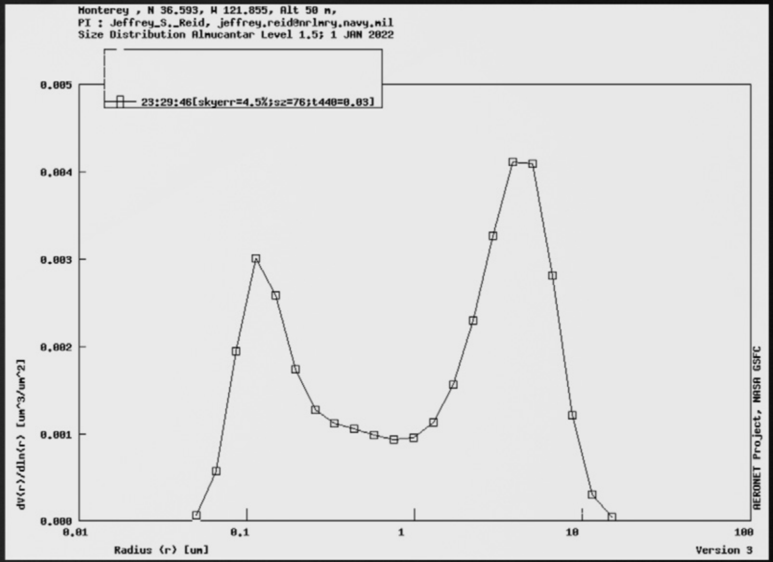


POLYPHONY™ DIGITAL

The vertical density profiles are given by multiple tables.

Modeling of the Aerosols

The Case Study

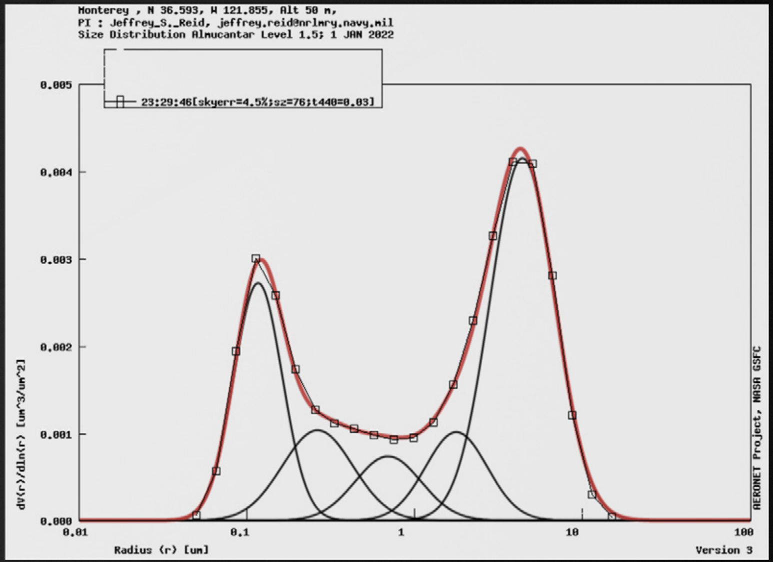


POLYPHONY™ DIGITAL

Let me show you an example, suppose the measured particle size distribution is like this.

Modeling of the Aerosols

The Case Study



POLYPHONY™ DIGITAL

It can be approximated by the sum of multiple log-normal distributions.

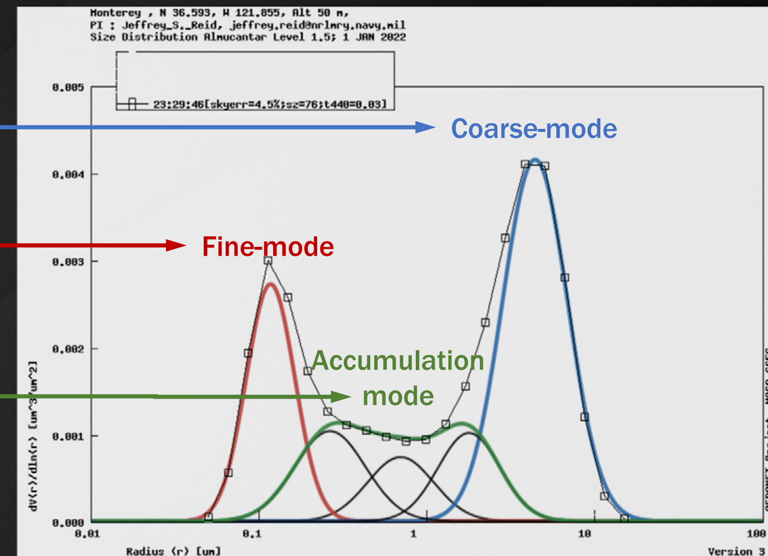
Modeling of the Aerosols

The Case Study

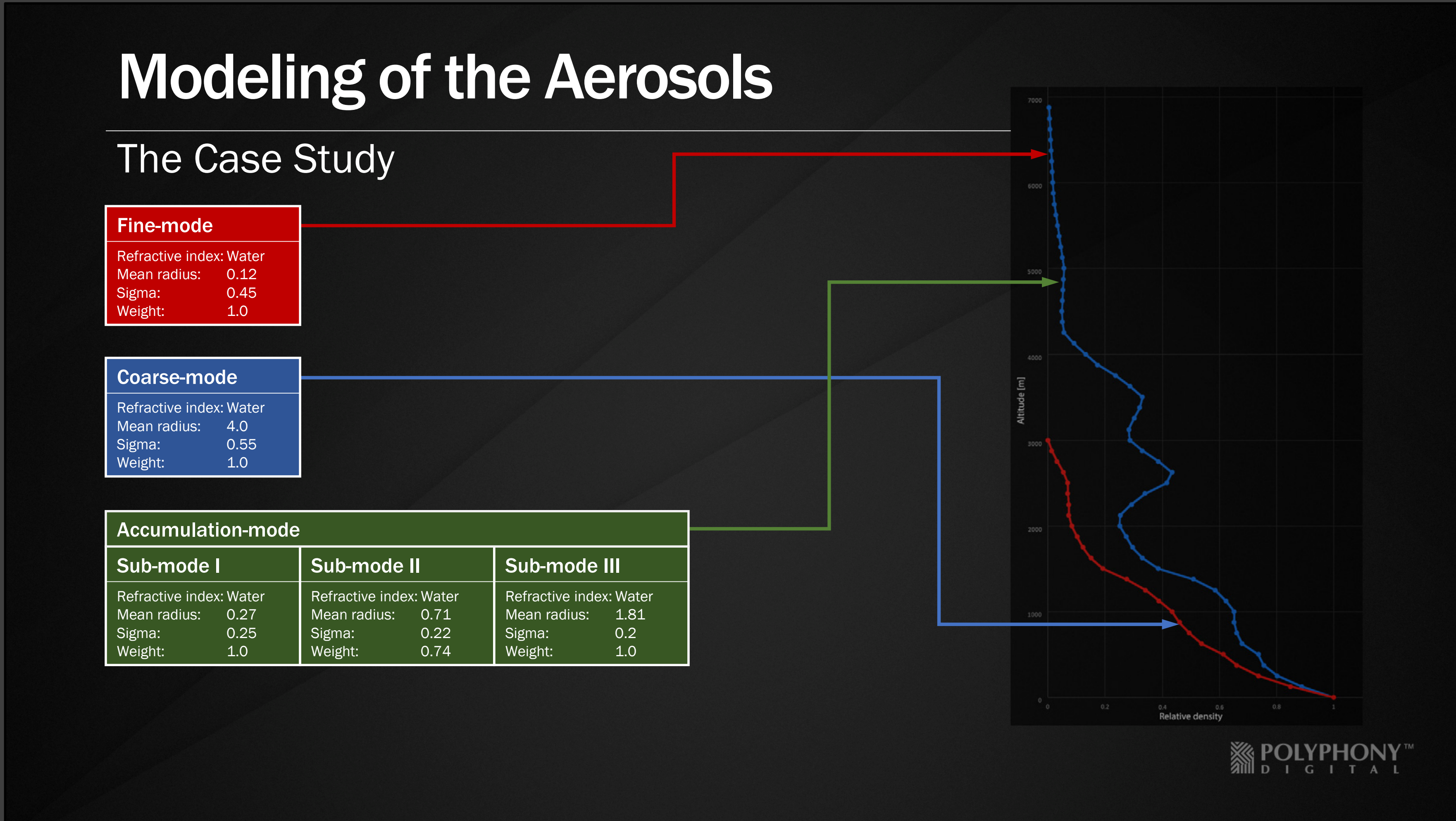
Fine-mode	
Refractive index:	Water
Mean radius:	0.12
Sigma:	0.45
Weight:	1.0

Coarse-mode	
Refractive index:	Water
Mean radius:	4.0
Sigma:	0.55
Weight:	1.0

Accumulation-mode					
Sub-mode I		Sub-mode II		Sub-mode III	
Refractive index:	Water	Refractive index:	Water	Refractive index:	Water
Mean radius:	0.27	Mean radius:	0.71	Mean radius:	1.81
Sigma:	0.25	Sigma:	0.22	Sigma:	0.2
Weight:	1.0	Weight:	0.74	Weight:	1.0



Each size distribution is defined and categorized as shown.



Then define the density profiles table for each aerosol.

As shown in this figure, several aerosols can share the same profile.

With this structure, we can describe the measurement data on the papers without missing anything.

Skysim

Implementation Detail

Next, we will discuss Skysim implementation.

Implementation Detail

Overview

- Offline Sky Renderer
  - C++, 20000 lines
- Features
  - Full spectrum path tracing
  - BHMIE for Mie theory solver
  - XYZ color matching function

POLYPHONY DIGITAL

An accurate atmosphere model requires an accurate renderer.

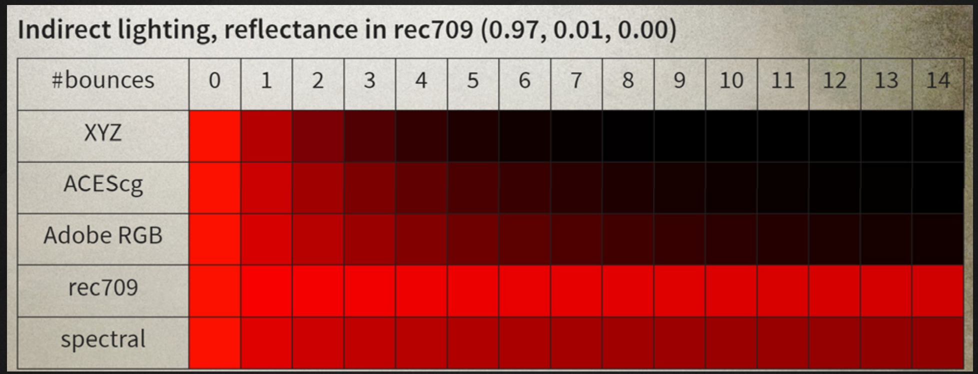
While most sky simulations are grid-based, Skysim uses path tracing; there would be no other choice for a reference renderer if we could ignore computation cost.

Mie scattering parameters are physically derived from the optical properties of each particle using the BHMIE algorithm.

Implementation Detail

Full Spectrum Path Tracing

- Error can be large with many reflections and scattering
  - Ordinally 3-channels renderings have a large error to spectral rendering (with many bounces)



https://jo.dreggn.org/path-tracing-in-production/2017/talk-jo/

POLYPHONY DIGITAL

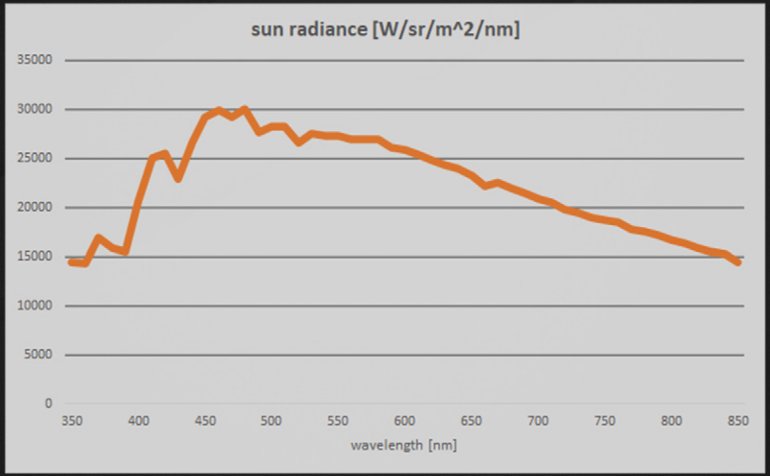
Skysim computes the intensities of each wavelength from the sky to avoid accumulating errors.

In general, rendering with many reflections and scattering must be done in the full spectrum otherwise the error can be large.

Implementation Detail

Full Spectrum Path Tracing

ASTM G173-03 Extraterrestrial Spectrum



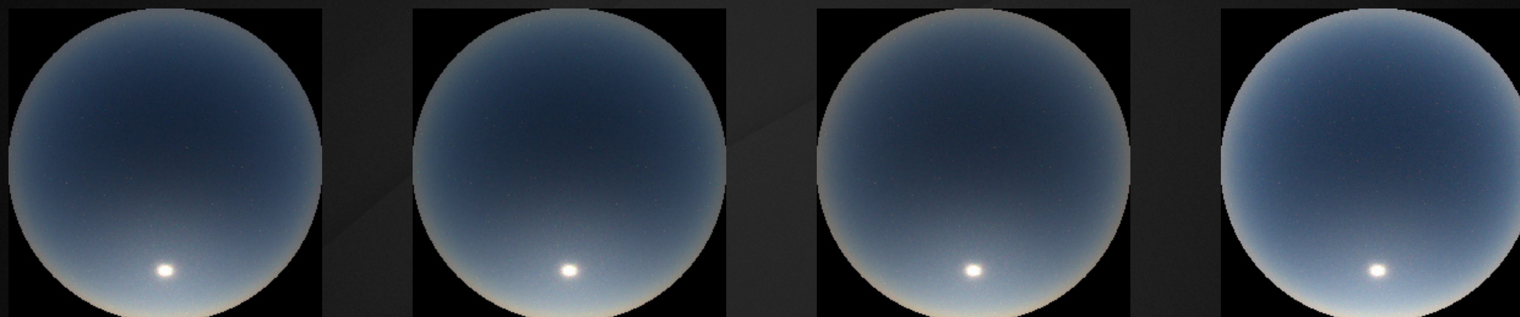
POLYPHONY™ DIGITAL

The solar spectral data is based on measured data from ASTM-Extraterrestrial radiation values but it was necessary to convert them to average radiance since they are given as irradiance.

While sunlight is approximated as a parallel light, it is a good enough approximation for our project.

Implementation Detail

Ground Albedo



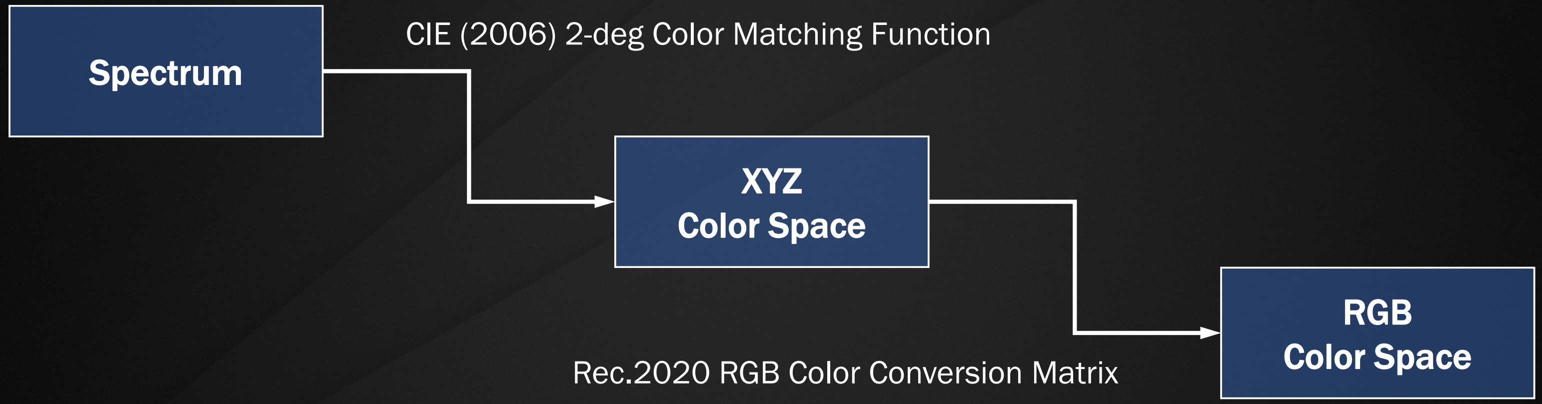
Oceans Forests Deserts Snows

Since Skysim is a full-spectrum renderer, RGB cannot be used for the ground albedo either, so it must be defined in terms of spectra.

Fortunately, it was easy to find many measurements in papers on global warming.

Implementation Detail

XYZ Color Matching Function



POLYPHONY™ DIGITAL

While the scattering simulation is performed as a full-spectrum energy calculation, the final rendering is RGB.

It is common practice to convert from spectral to XYZ and then convert to RGB, the problem can be simplified by separating the transformation between each color space via XYZ.

Implementation Detail

Other Algorithm Optimization

- Ratio-tracking to estimate transmittance functions [Novák 2014]
- Importance sampling
  - The distribution of phase functions
  - Wavelengths using spectral luminous efficiency
- Denoiser
- Adaptive sampling for near-sun angles
- etc.

The computation of the volume rendering equation with path tracing is quite expensive.

Ratio-tracking, some importance sampling methods, denoiser, adaptive sampling, and so on.

Implementation Detail

Other Optimization

- Profile-based optimization
  - 10% reduction in computation time
- Per-pixel multi-threading support
- Distributed rendering using multiple PCs

POLYPHONY™ DIGITAL

We use other kinds of optimizations.

Profile-based optimization is very useful. It’s easy to apply, and this reduces 10% in computation time.

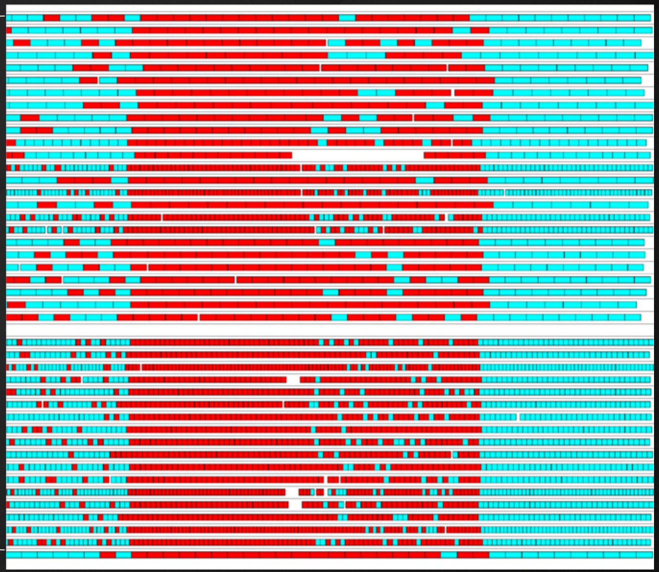
Of course, per pixel threading is supported.

Implementation Detail

Other Optimization

- Distributed rendering using multiple PCs

1 column = 1 PC for 1 development staff



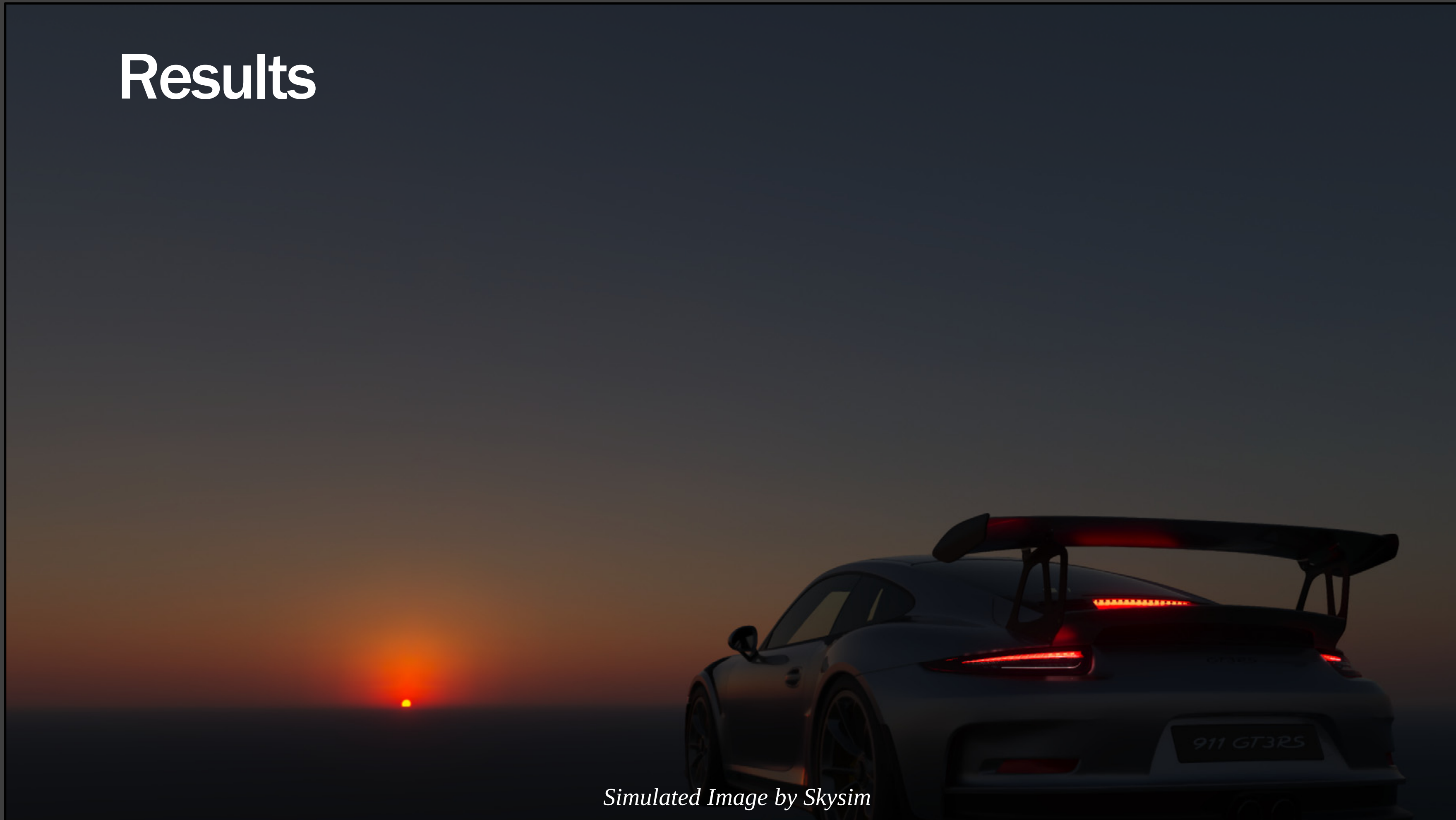
In addition, we distribute rendering tasks using multiple PCs. We made a system that allows the free use of any available PC in our studio.

Skysim

Results

Thank you for your patience, it is finally time to show you final images!

Results



Simulated Image by Skysim

Yes, we've got what we desired.

Results



Simulated Image by Skysim

Notice the light spreading out over the sun.

Results



Simulated Image by Skysim

No loss of contrast even in haze.

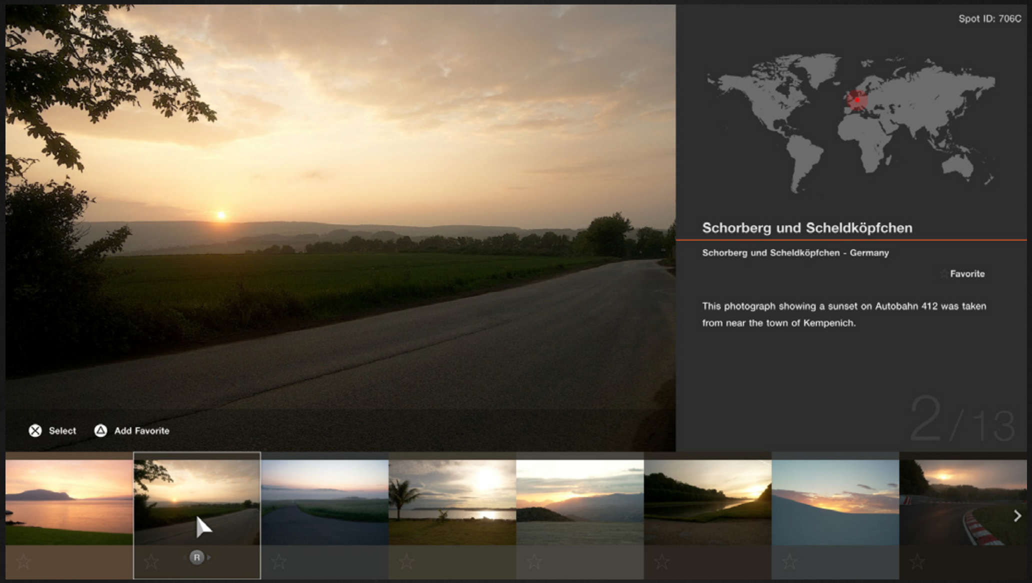
Skysim

Validation with Scapes

It took a lot of hard work for a year, but we were lucky to have Scapes in Gran Turismo.

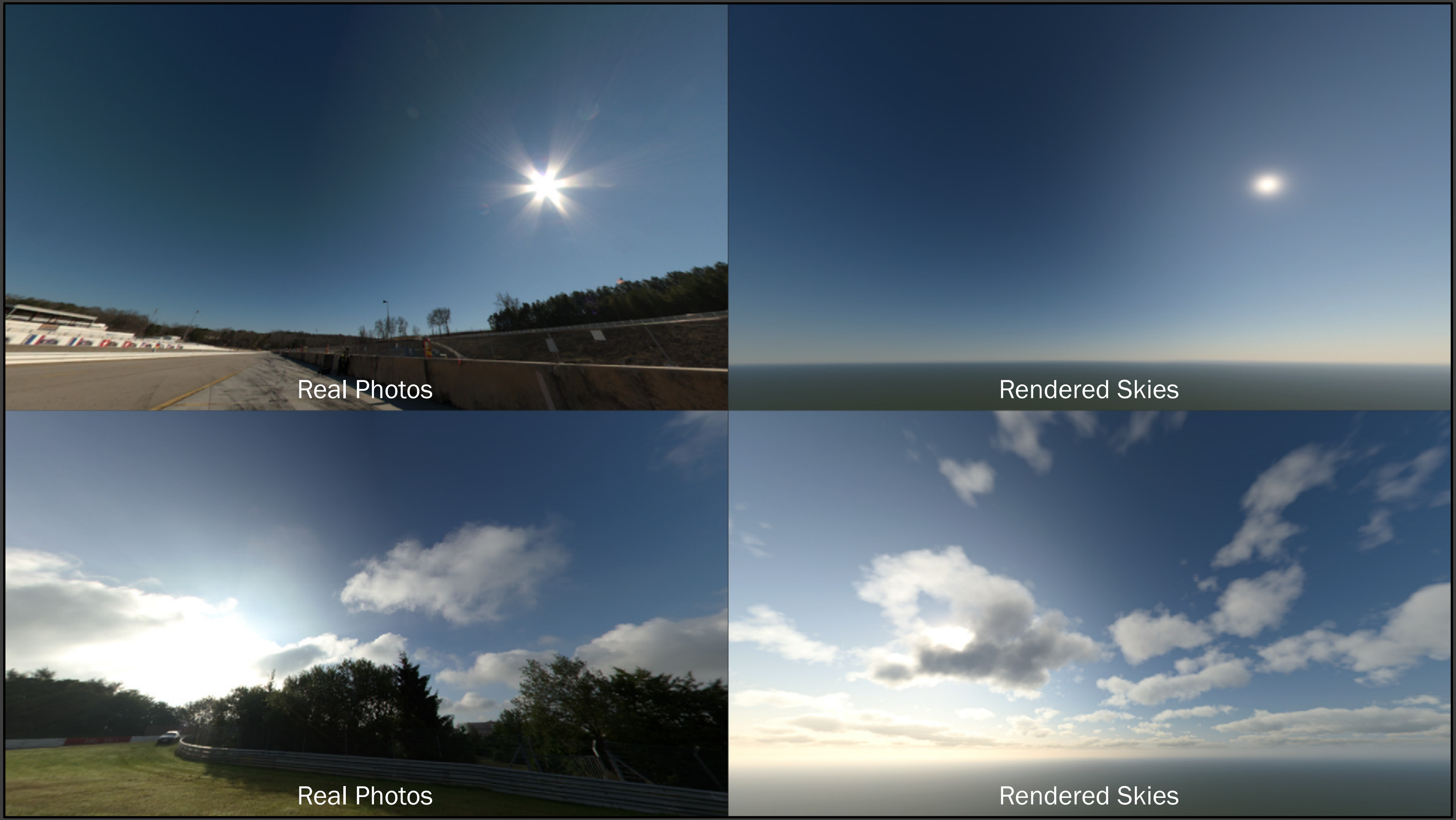
Validation with Scapes

What is Scapes?



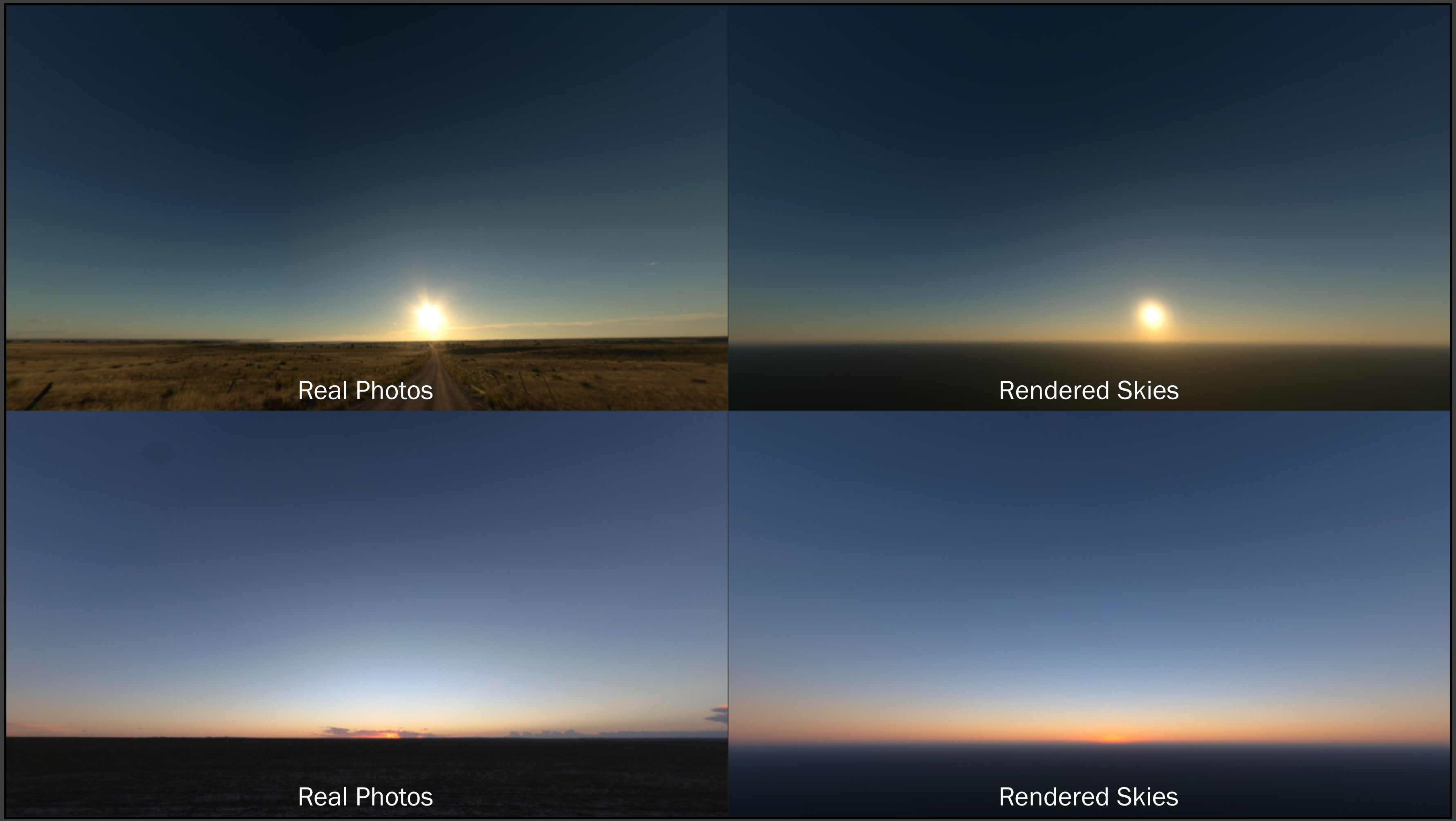
Scapes is one of the Gran Turismo content that has thousands of HDR panoramic photos with IBL information.

We could judge the quality of the simulation whether the sky is similar enough to the photo.

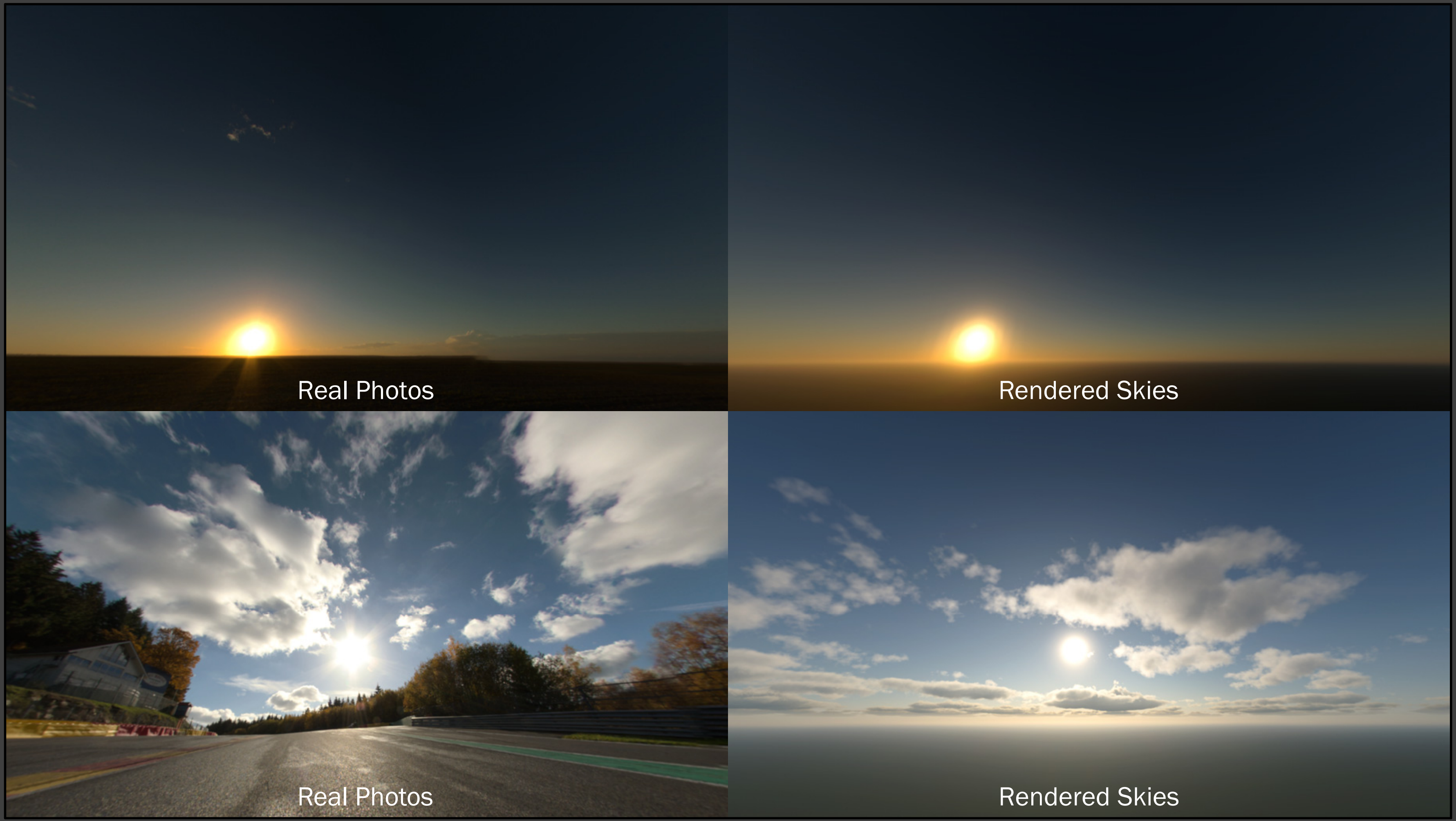


On the left is the actual photo and on the right is the rendering result.

Although there is a cloud in the images that we haven't explained yet, please let it slide.



Many details of the aerosols are still unknown today, and the available measurements are always incomplete.



Much of the missing data had to be guessed from the few clues, and it was a constant process of rendering and modifying.

Actually, it was the Physically-Based Look Development.



Thanks to the Scapes, we were able to avoid too much artistic tweaking.

We would never have finished this experiment without Scapes.



Oh, this is an experiment and I have to write a review.

Review

- Visualizations were quite satisfactory
- Rayleigh scattering was pretty accurate
- Mie scattering of the aerosol was full of problem
- The contribution of each accuracy has not been verified in detail

POLYPHONY™ DIGITAL

Most of the frustrations of the existing simulations had been eliminated, so I would say it was a success.

It is obvious that the accuracy of Mie scattering is a problem.

However, everything was as accurate as possible, the contribution of each accuracy was not verified in detail.

And…

Review

- Visualizations were quite satisfactory
- Rayleigh scattering was pretty accurate
- Mie scattering of the aerosol was full of problem
- The contribution of each accuracy has not been verified in detail
- **The computation cost was incredible.**

POLYPHONY™ DIGITAL

Yabai desu. *Note: Means “incredible” or “terrible” in Japanese.

Even with distributed rendering, it was like ray-tracing at home in the 90s, it was badly difficult to predict the rendering time.

Skysim

Extra Result

In addition, let's talk about something suitable for racing games.

There was a positive effect that we didn’t expect.

Extra Result



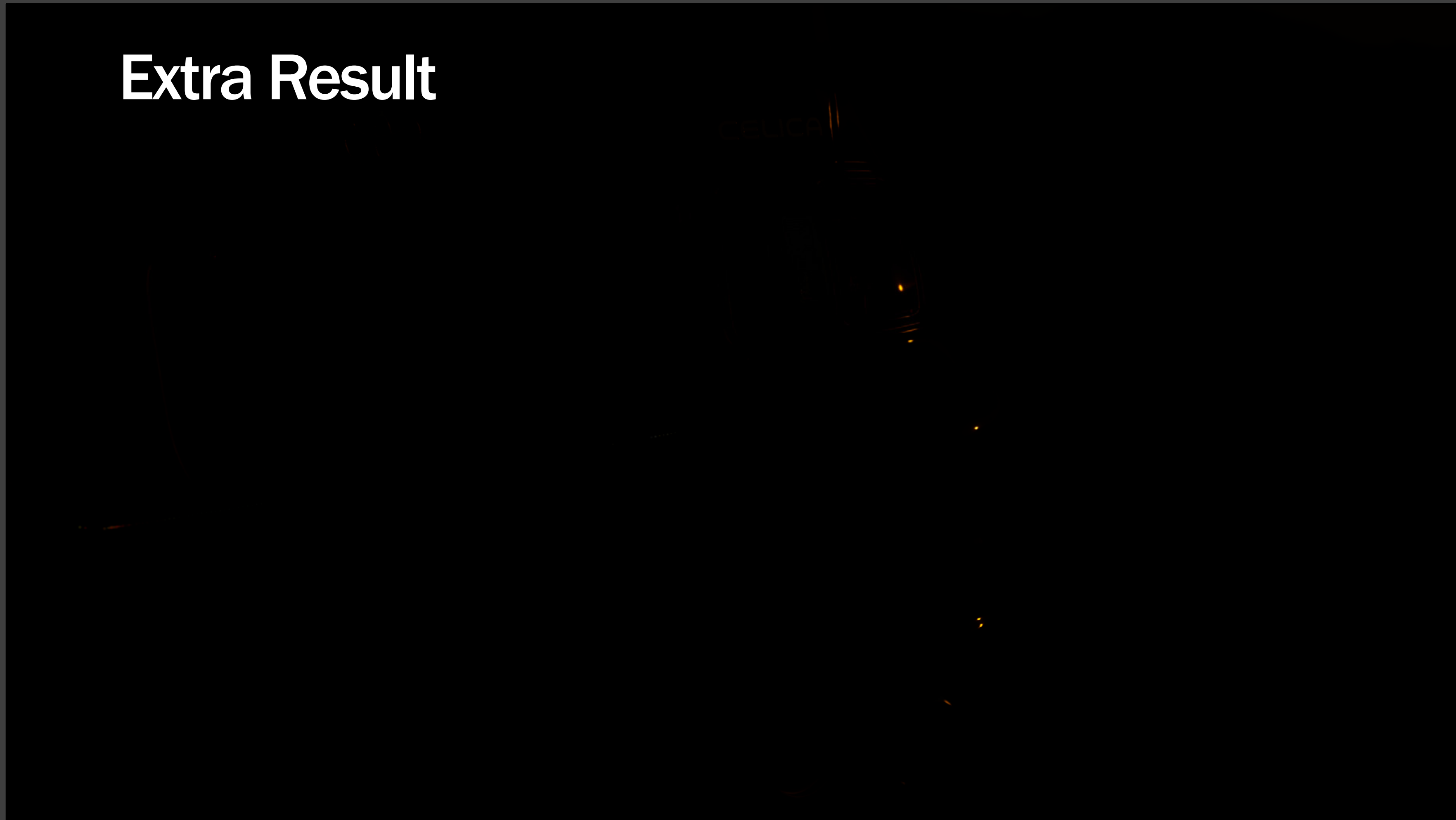
Strong forward scattering of the sun naturally makes the specular larger.

Extra Result



Let's get closer.

Extra Result



Actual size of the sun is like this.

Extra Result



Not only size but has a nice anti-aliasing blur that differs from both GGX and post-effects Gaussian.

Extra Result



In Gran Turismo Sport, we sometimes rendered the specular larger to address specular under-sampling.

However, it was easy to lose the texture and scale, the distorted and bleached sun is not cool.

Extra Result



Using roughness to blur the sun makes it too dull and looks like a poorly cared car.

Extra Result



Hmm, sexy, isn't it?

You think I'm a bit weird?

Extra Result



Anyway, the contrast of the sky is often good for cars.

OK, that's all for my part, let's switch to Kentaro's turn.

Agenda

- Introduction

- Physics of Sky
- Simulation of Sky
- Real Earth Atmosphere (and Simulations)
- Skysim --- Our Sky Renderer

- Run-time Sky Rendering Method
- Run-time Cloud Rendering Method
- Summary

POLYPHONY™ DIGITAL

Run-time Sky Rendering Method

POLYPHONY™ DIGITAL

Hello, everyone, My name is Kentaro Suzuki and I'm a graphics engineer.

From this point on, I will give my presentation.

Next I will talk about the runtime method of our sky rendering.

Run-time Implementation Concepts

- Our “off-line” sky renderings
  - High quality
  - High computation cost
- We wanted to use these results “at runtime”

POLYPHONY™ DIGITAL

Let me first explain the concept.

As we have discussed, we are confident that the quality of sky rendering was greatly improved by the accurate atmospheric modeling and scattering simulation.

Of course, we wanted to achieve this quality at run-time and in-game. However, such amount of computation cannot be done at runtime.

Run-time Implementation Concepts

- Our “off-line” sky renderings
  - High quality
  - High computation cost
- We wanted to use these results “at runtime”

Look-up Table ☺

POLYPHONY™ DIGITAL

Therefore, we decided to use the results of skysim as a look-up table. Using the look-up table, we can get high quality skies to get values from textures.

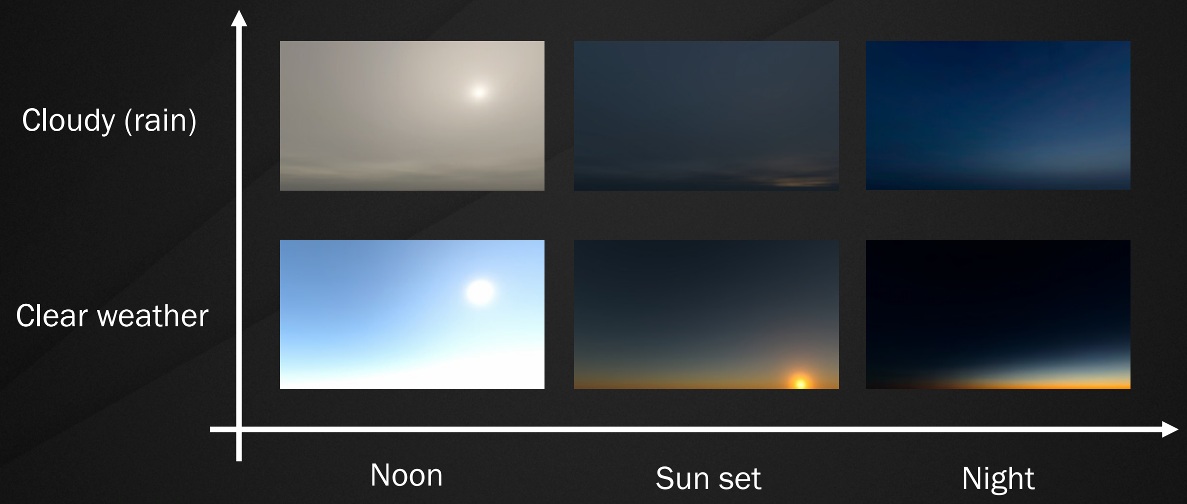
Look-up Table

- GT7 supports dynamic time and weather

In Gran Turismo 7, we supports dynamic time and weather change features.

Look-up Table

- GT7 supports dynamic time and weather
  - We rendered sky images in two axes, time and weather for look-up tables
  - No altitude changes



POLYPHONY DIGITAL

So we rendered sky images in two axes, time and weather, and converted them into look-up tables.

The altitude change in Gran Turismo 7 is about 300m, so it can be excluded from the Look-up Table parameters.

Single Look-up Table Texture

- 3D LUT texture stores sky radiance
  - Retaining pre-computed high quality sky rendering results
  - Corresponds to a single weather condition
  - 1D is for time component and 2D is for spherical direction

Our Look-up Table is a 3D texture that stores the radiance from the sky in all spherical directions. This allows us to maintain high quality pre-computed sky rendering results.

A single 3D texture corresponds to a single weather condition.

Single Look-up Table Texture

- 3D LUT texture stores sky radiance
  - Retaining pre-computed high quality sky rendering results
  - Corresponds to a single weather condition
  - 1D is for time component and 2D is for spherical direction

- Texture resolutions (adaptively mapped, sunny weather LUT)

Time (sun elevation)	64 slices
Spherical direction	32 px * 128 px
Texture format	R11F_G11F_B10F * BC6h is not sufficient for color accuracy

POLYPHONY DIGITAL

We adaptively map one dimension in the time component and two dimensions in the spherical direction. These adaptive mappings save a lot of texture resolution.

This figure shows one example of resolution and format of the texture.

Run-time Sky Rendering Method

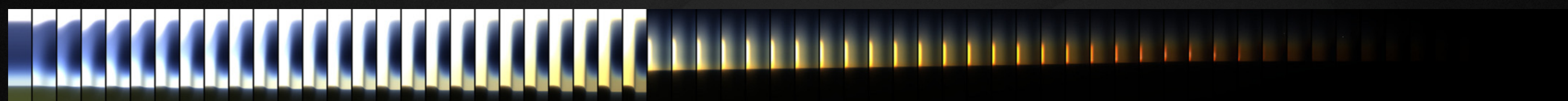
Mapping of Look-up Table Time Component & Spherical Direction

I will now talk about the mapping of the LUT with respect to time component and spherical direction.

Adaptive Mapping of LUT (time component)

- LUT is a 2D texture array of 64 slices

Time (sun elevation), 64 two-dimensional textures

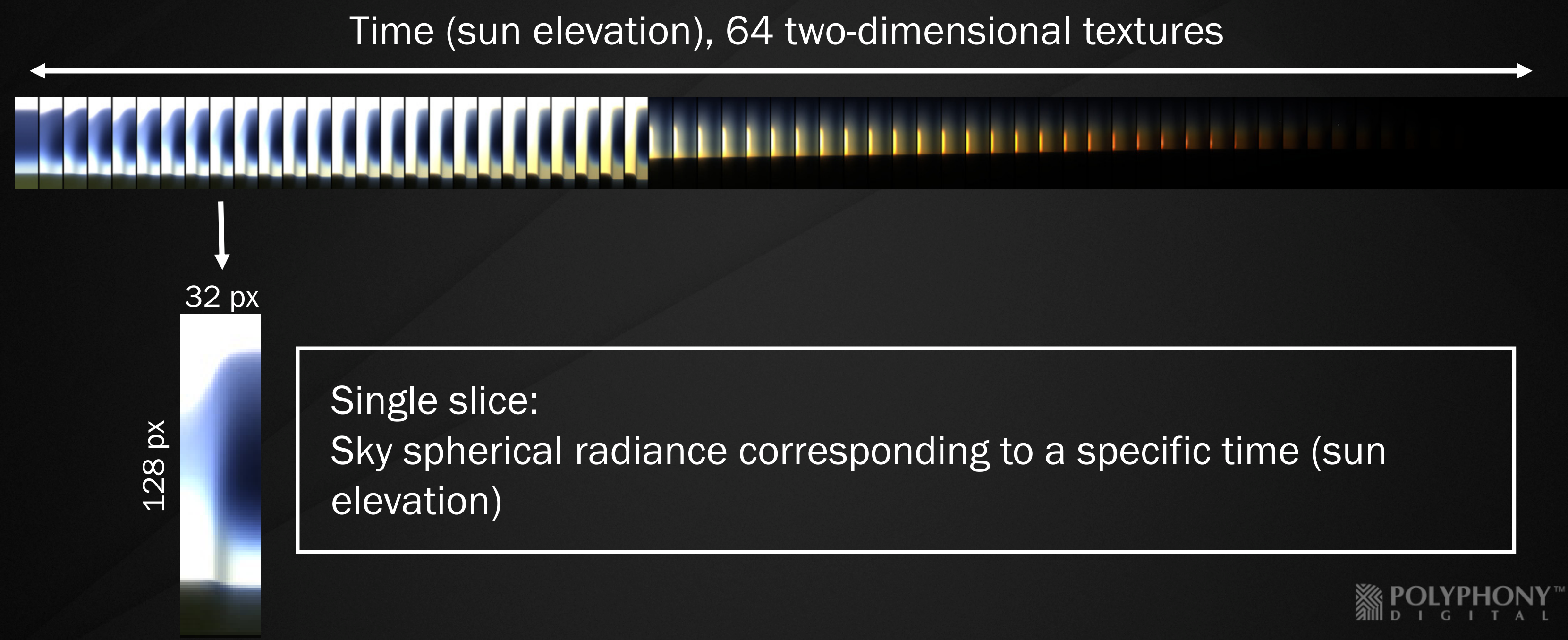


First, I will explain the mapping of time. An example Look-up Table is shown here.

64 two-dimensional textures are lined up in the time direction.

Adaptive Mapping of LUT (time component)

- LUT is a 2D texture array of 64 slices

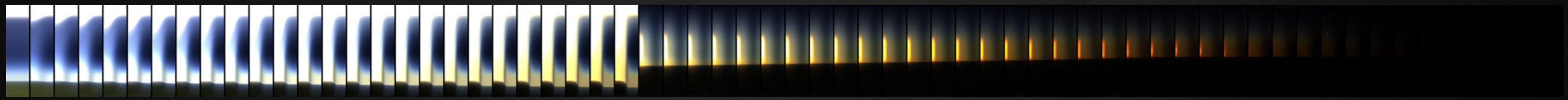


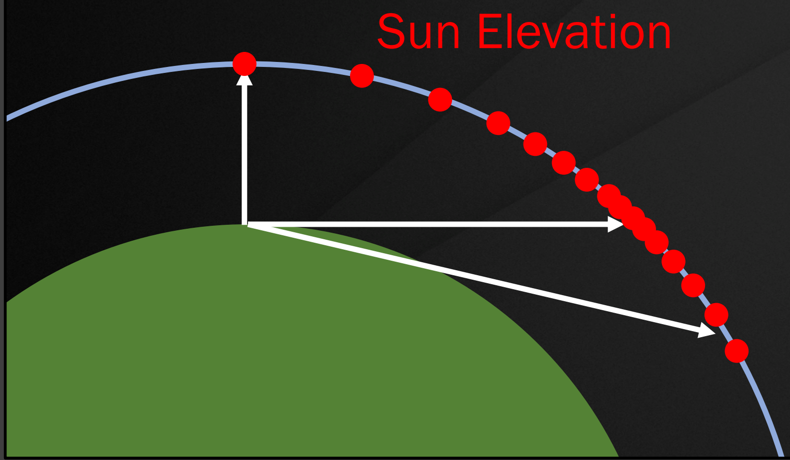
POLYPHONY DIGITAL

Each slice contains the sky spherical radiance corresponding to a particular time.

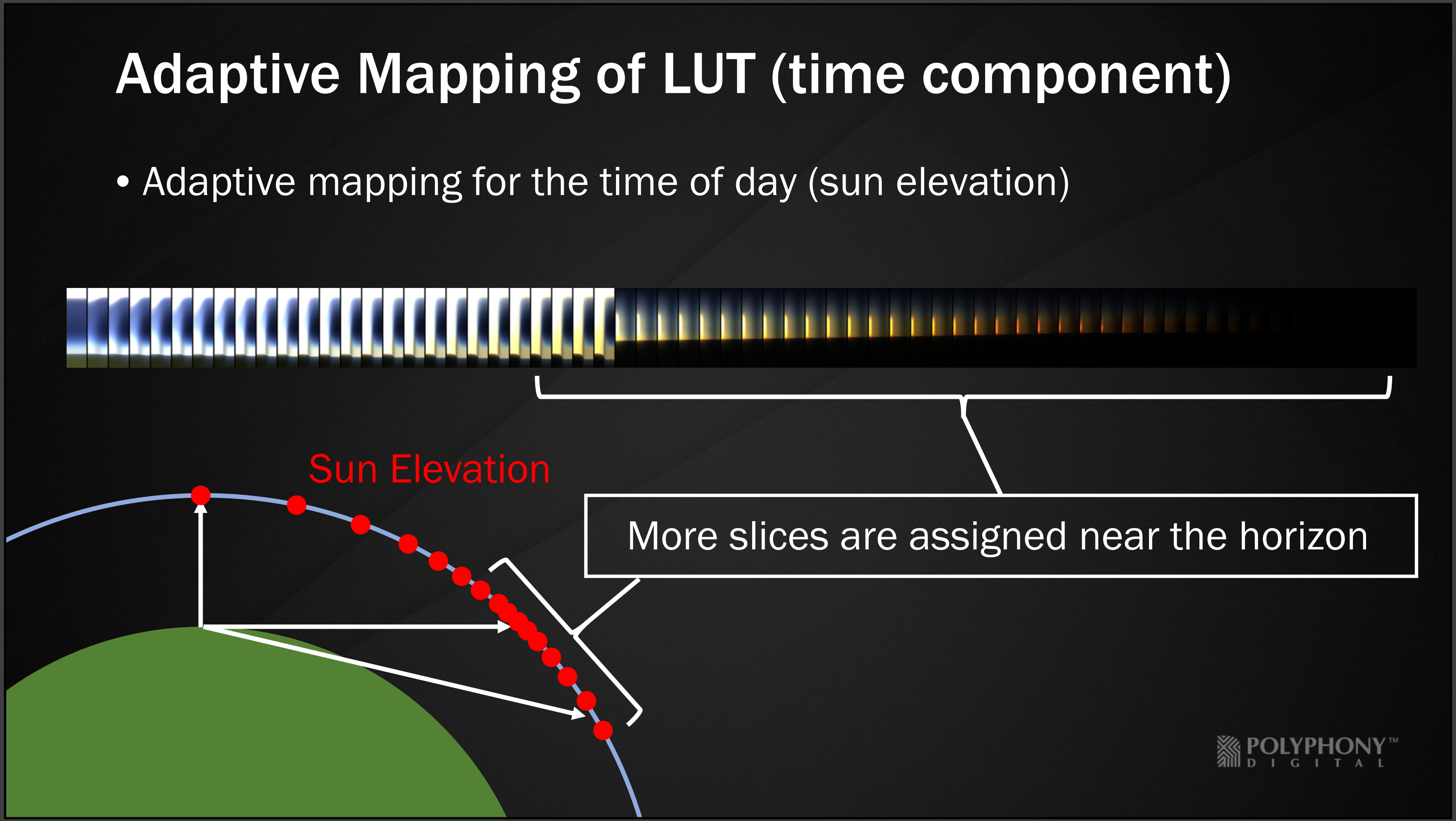
Adaptive Mapping of LUT (time component)

- Adaptive mapping for the time of day (sun elevation)





To use textures efficiently, each slice is adaptively mapped to the time of day. Here, it corresponds to sun elevation.

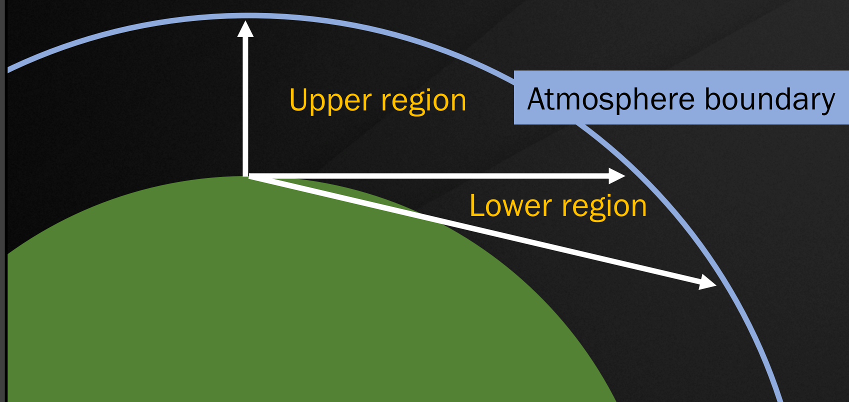


As you can see, more textures are assigned for altitudes near the horizon.

This is because the sky luminance changes significantly around the sunset.

Adaptive Mapping of LUT (time component)

- Upper region
  - 48 slices
- Lower region
  - 16 slices
  - Limited due to the minimum sun elevation angle (-10 [deg])
    - Linear interpolation toward a constant value (< -10 [deg])



POLYPHONY DIGITAL

I will explain more about adaptive mapping for the time component.

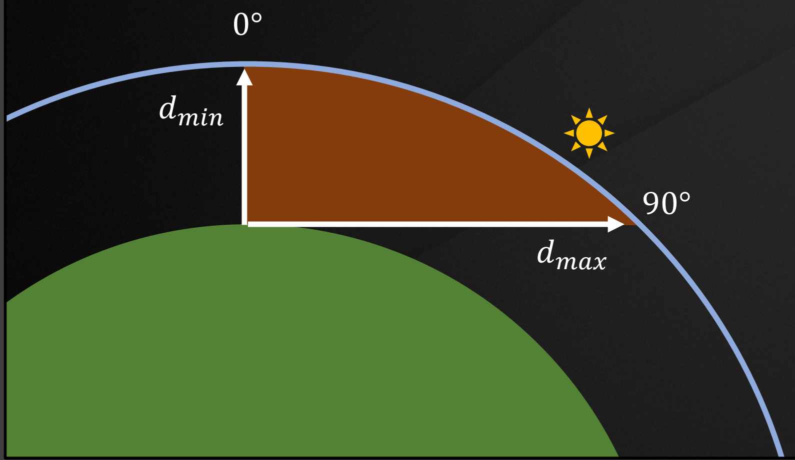
We divide the cases according to whether the sun is in the region above or below the horizon.

We assigned 48 slices to the upper region and 16 to the lower region.

Lower regions is limited due to the minimum sun elevation angle (-10 [deg])

Adaptive Mapping of LUT (time component)

- Upper region
  - d_{min} and d_{max} are the vertical and horizontal distances to the atmospheric boundary.



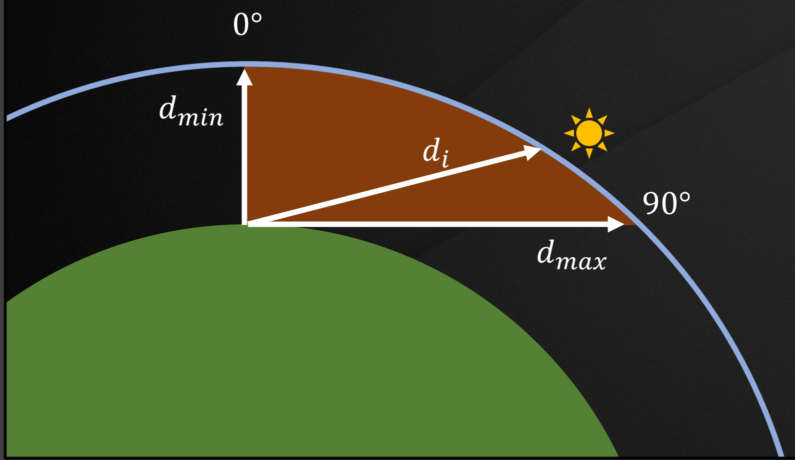
POLYPHONY DIGITAL

This is the mapping in the upper region.

d_{min} and d_{max} are the vertical and horizontal distances to the atmospheric boundary.

Adaptive Mapping of LUT (time component)

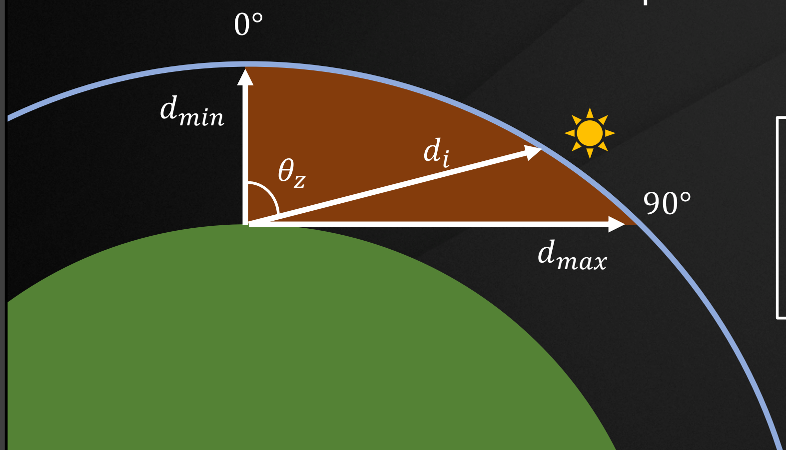
- Upper region
  - d_{min} and d_{max} are the vertical and horizontal distances to the atmospheric boundary.
  - $d_i = \mathrm{lerp}\left(d_{min}, d_{max}, \frac{sliceIndex}{47}\right)$



From these parameters, d_i can be calculated with respect to the slice index using lerp function.

Adaptive Mapping of LUT (time component)

- Upper region
  - d_{min} and d_{max} are the vertical and horizontal distances to the atmospheric boundary.
  - $d_i = \text{lerp}\left(d_{min}, d_{max}, \frac{sliceIndex}{47}\right)$
  - Zenith distance of sun θ_z is decided from d_i
  - Each LUT slice is computed from θ_z



More slices are assigned when the sun is near the horizon
(Sky changes mainly depend d_i)

POLYPHONY DIGITAL

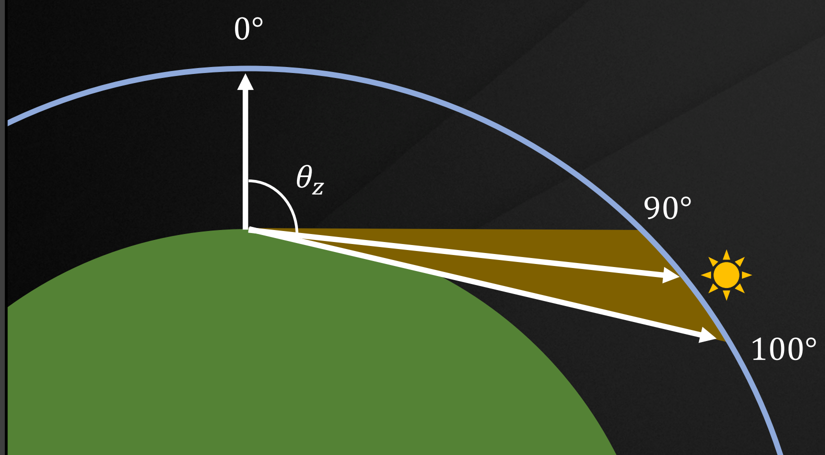
And once d_i is determined, we can obtain the zenith distance of sun θ_z.

Finally, each LUT slice is calculated from θ_z

Using this technique, more slices are assigned when the sun is near the horizon.

Adaptive Mapping of LUT (time component)

- Lower region
  - $\theta_z = \mathrm{lerp}\left(90°, 100°, \left(\frac{sliceIndex - 48}{15}\right)^2\right)$
  - Each LUT slice is computed from θ_z

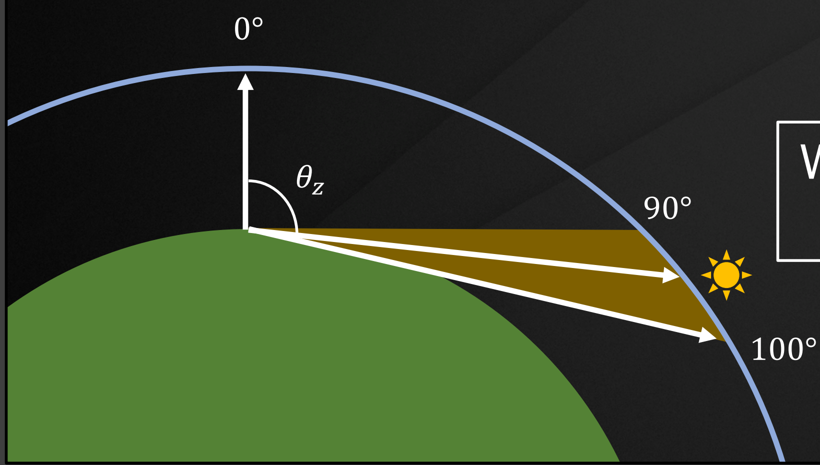


This is the mapping in the lower region.

θ_z is calculated from the simple interpolation and each LUT slice is calculated from θ_z

Adaptive Mapping of LUT (time component)

- Lower region
  - $\theta_z = \text{lerp}\left(90°, 100°, \left(\frac{sliceIndex - 48}{15}\right)^2\right)$
  - Each LUT slice is computed from θ_z



When the sun is near the horizon, more slices are assigned, too.

As in the upper region, more slices are assigned when the sun is near the horizon.

Adaptive Mapping of LUT (spherical direction)

- Adaptive LUT mapping for spherical direction
  1. Interpolation between different LUT (time) slices is possible
  2. Mapped around the sun's direction, so it can be decoded as a sky for any sun elevation
  3. More texels around the sun and in the horizon direction (important regions)

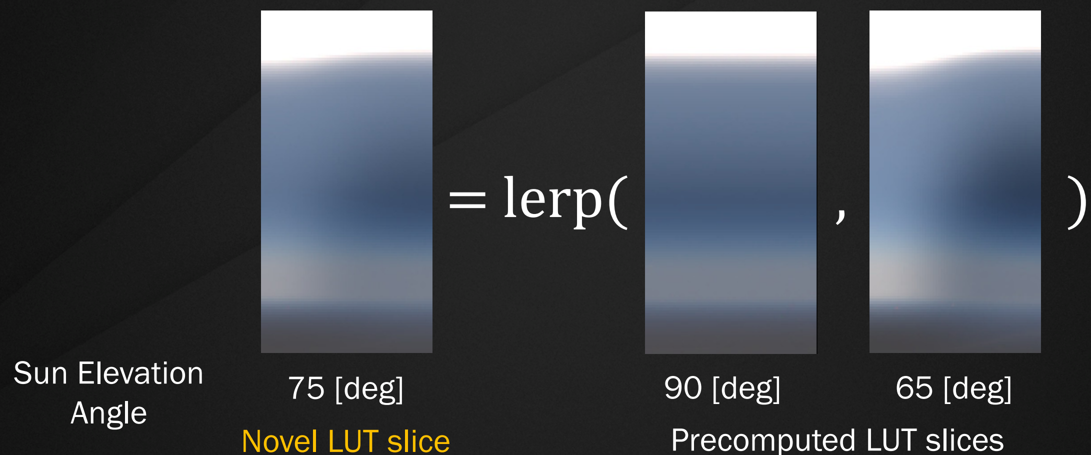
Next, let me talk about the adaptive mapping for the spherical direction.

This mapping has several features. I will explain them shortly.

Adaptive Mapping of LUT (spherical direction)

- Adaptive LUT mapping for spherical direction
  1. Interpolation between different LUT (time) slices is possible

It is possible to interpolate two different LUT slices
(e.g. 90[deg] and 65[deg] solar elevation angle)



First, with this mapping, it is possible to interpolate two different LUT slices smoothly.

As you can see, two slices share the texture space parameterizations, so interpolating between two precomputed slices is easy.

In this way, we can get any LUT slice of sun elevation at runtime.

Adaptive Mapping of LUT (spherical direction)

- Adaptive LUT mapping for spherical direction
  1. Interpolation between different LUT (time) slices is possible
  2. Mapped around the sun’s direction, so it can be decoded as a sky for any sun elevation

It is possible to decode interpolated LUT slice as a sky for any sun elevation

POLYPHONY™ DIGITAL

It is not enough to be able to interpolate two slices.

In addition, with this mapping it is possible to decode the interpolated LUT slice as a sky for any sun elevation.

Adaptive Mapping of LUT (spherical direction)

- Adaptive LUT mapping for spherical direction
  1. Interpolation between different LUT (time) slices is possible
  2. Mapped around the sun's direction, so it can be decoded as a sky for any sun elevation

➡ Smooth time transition of sky using sparse LUT slices

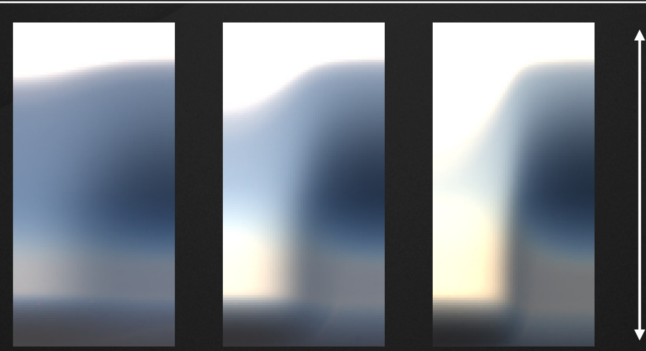
POLYPHONY™ DIGITAL

With these features, we can achieve a smooth temporal transition of the sky using sparse LUT slices.

Adaptive Mapping of LUT (spherical direction)

- Adaptive LUT mapping for spherical direction
  1. Interpolation between different LUT (time) slices is possible
  2. Mapped around the sun's direction, so it can be decoded as a sky for any sun elevation
  3. More texels around the sun and in the horizon direction (important regions)

Texture resolution can be saved while maintaining quality

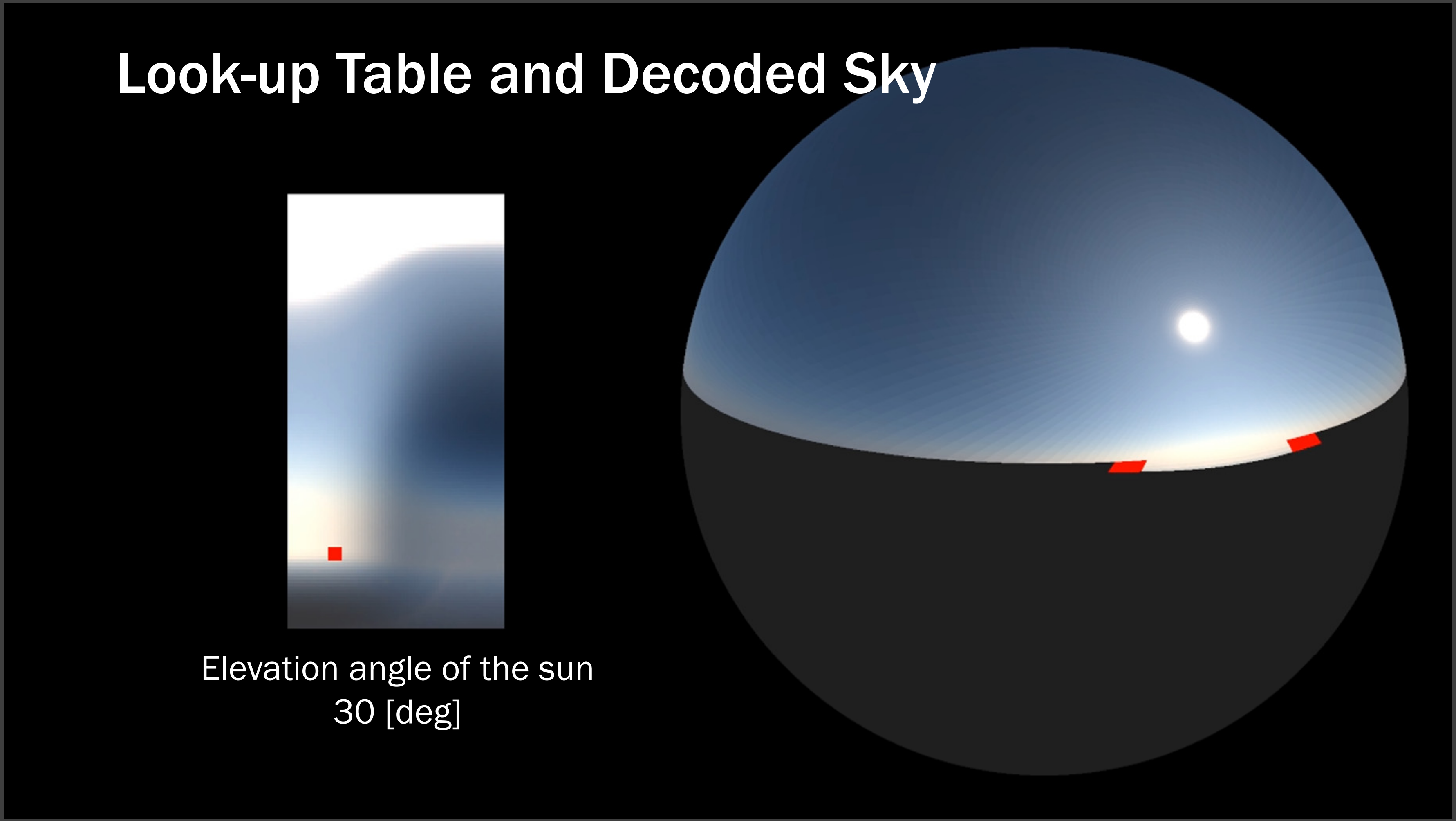


POLYPHONY™ DIGITAL

Apart from these features, adaptive mapping for texture resolution is also implemented.

We assign more texels around the sun and towards the horizon.

This allows us to save texture resolution while maintaining quality.

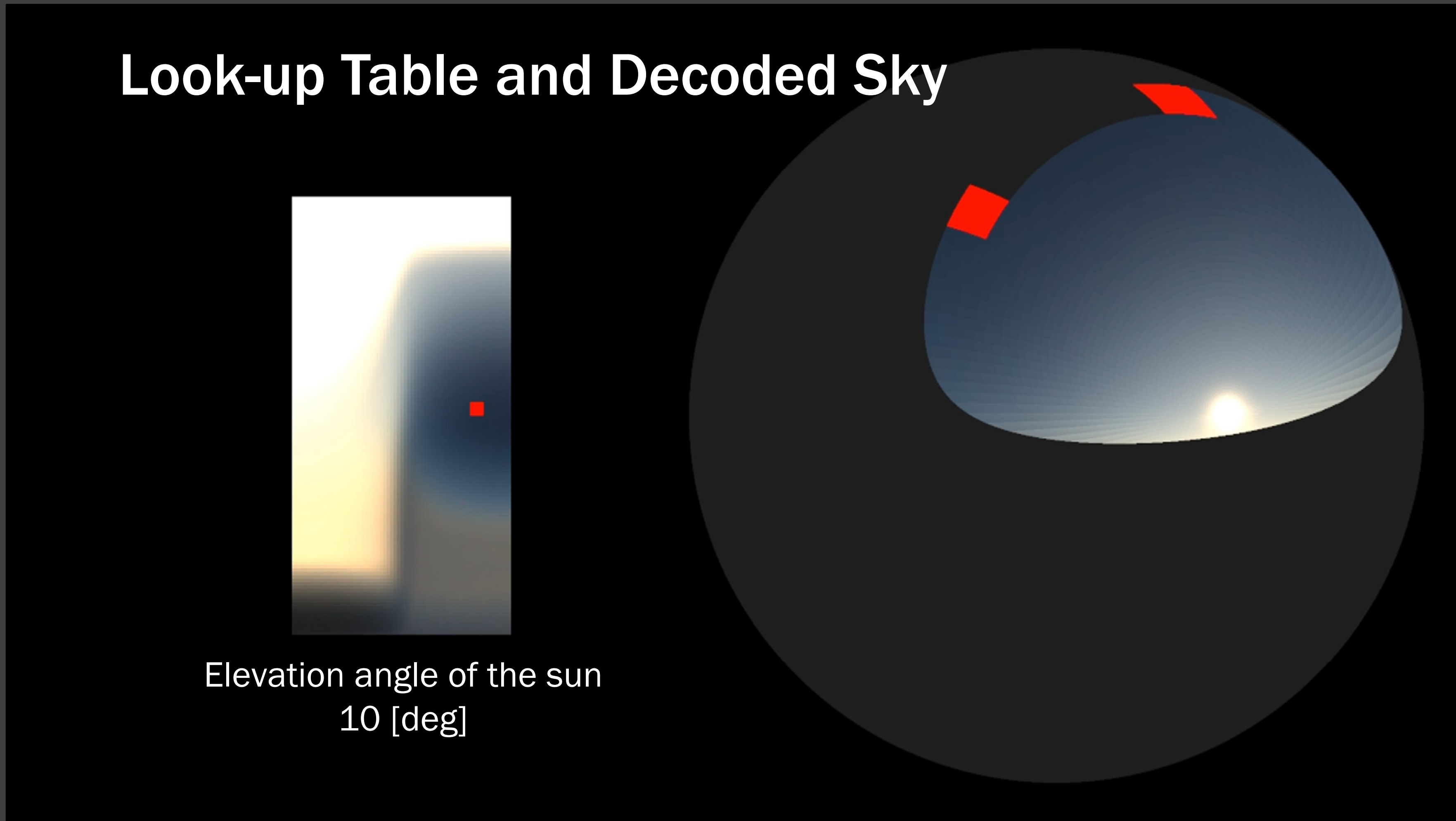


(video)

The video shows how the LUT is mapped to the sky, and how the sky is decoded.

The left image is the LUT slice and the right image is the decoded sky. The red areas on the left and right correspond.

As you can see, there is more texel density around the sun and near the horizon. Also, the overall mapping is very nonlinear.



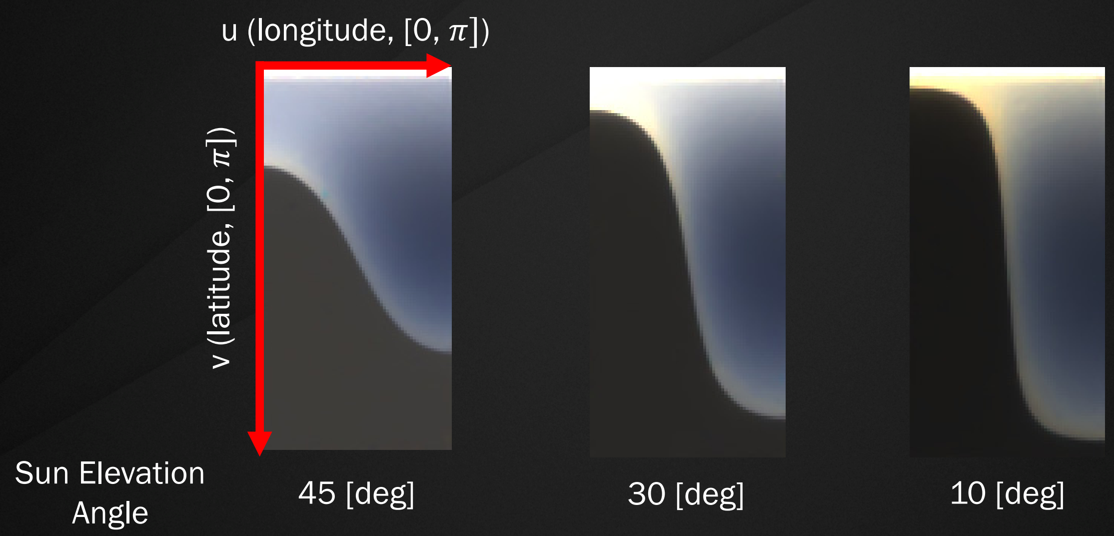
(video)

This is another example.

You can see that the mapping changes from the previous one depending on the elevation of the sun.

Detail of Adaptive Mapping (spherical direction)

- First version
  - Simple latitude and longitude sky mapping (equirectangular mapping)
  - The direction of the sun was the upward direction

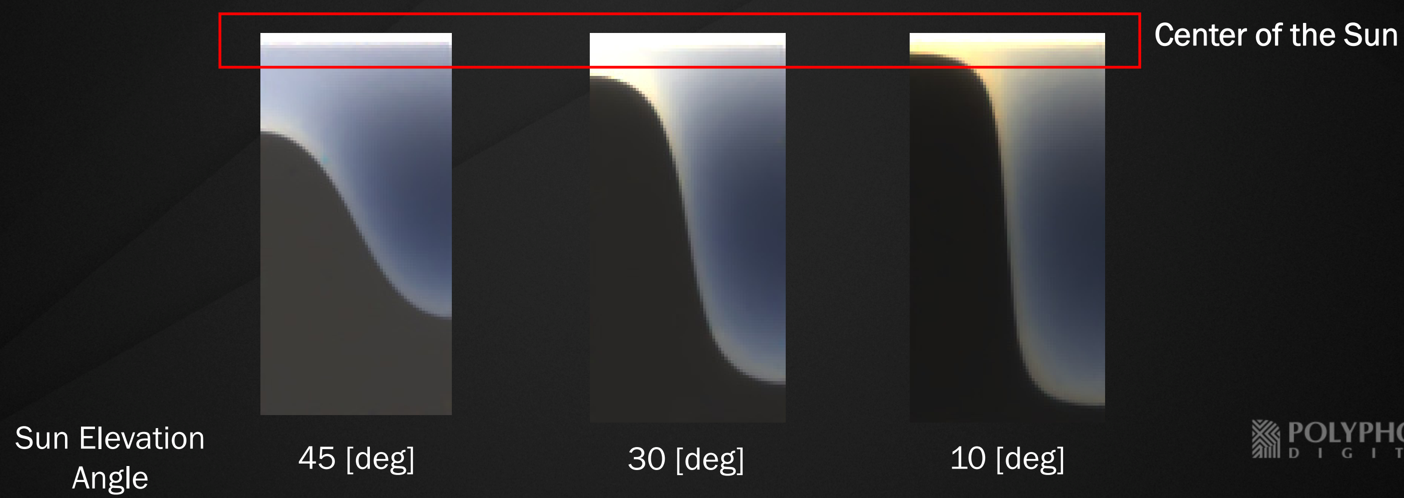


Let me explain the details of this spatially adaptive mapping. I will explain about our first version mapping.

In the first version, we tried a simple latitude and longitude sky mapping, so called equirectangular mapping, but the direction of the sun was the upward direction.

Detail of Adaptive Mapping (spherical direction)

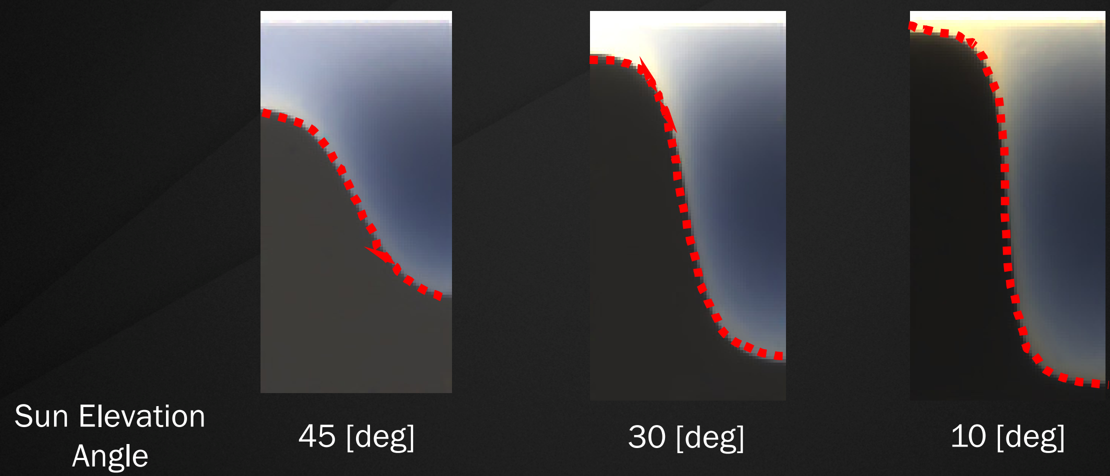
- First version
  - Simple latitude and longitude sky mapping (equirectangular mapping)
  - The direction of the sun was the upward direction



Because the sun was the upward direction, the position of the sun was aligned in the texture slices.

Detail of Adaptive Mapping (spherical direction)

- Problems:
  1. Low interpolation quality due to the unaligned horizon in the LUT
  2. Distorted horizon in resulting skies



POLYPHONY™ DIGITAL

However, there were several problems.

We tried to interpolate between the LUT slices by aligning the sun, but the interpolation quality of the horizon was poor because the horizon was not aligned.

In addition, the final resulting horizon was distorted because the horizons in the LUT were distorted.

This is indicated by the red dotted lines in the images.

Detail of Adaptive Mapping (spherical direction)

- When rendering the sky with these LUTs, the horizon line became dirty

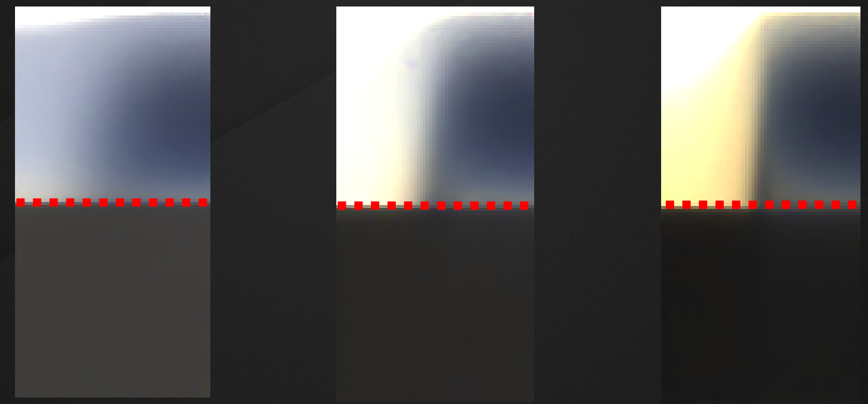


When rendering the sky with these LUTs, the horizon line became distorted.

This was not acceptable to us.

Detail of Adaptive Mapping (spherical direction)

- Second version (with normalization)
  - Calculated the angle from the sun to the horizon and use this to normalize in the direction of latitude
  - This makes the horizon a straight line on the LUT ☺
    - Sun and horizon are aligned



POLYPHONY™ DIGITAL

So as a second version we introduced normalization.

We calculated the angle from the sun to the horizon and used that to normalize in the direction of the latitude.

This approach made the horizon a straight line on the LUT.

Using this normalization, we can interpolate two different slices very well because sun and horizon are aligned.

Detail of Adaptive Mapping (spherical direction)

- It also result in a straighter horizontal line on the rendered image

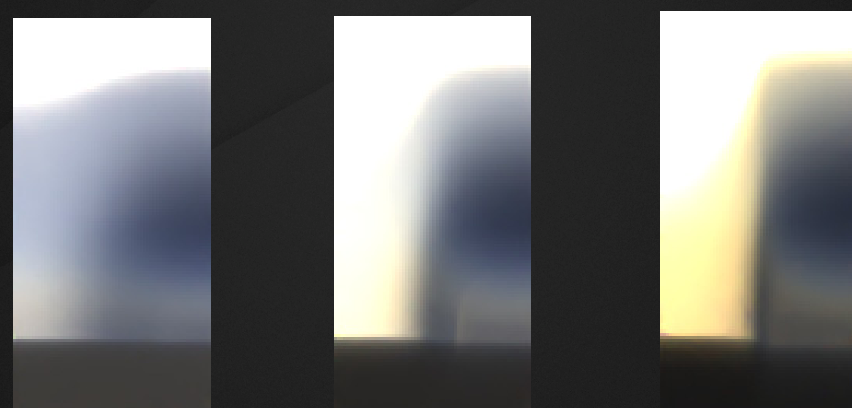


And this normalization resulted in a straighter horizontal line on the rendered image.

This is a more desirable result.

Detail of Adaptive Mapping (spherical direction)

- Nonlinear transformation of the v-coordinates
  - Important sky regions
    - Above horizon
    - Around the sun
- This transformation increased the amount of important texels in the LUT



Sun Elevation Angle | 45 [deg] | 30 [deg] | 10 [deg]

POLYPHONY DIGITAL

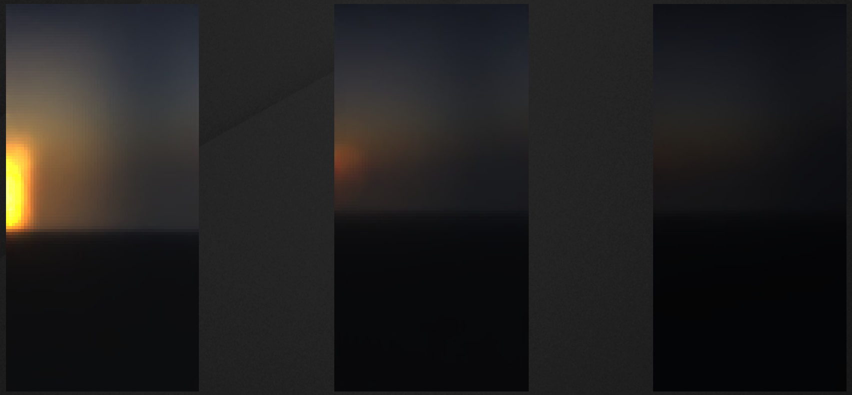
In addition to the previous method, we further nonlinearly transformed the v-coordinates.

Since the final important regions are the upper horizon and the regions around the sun, we designed a transformation that takes them into account.

This increases the amount of important texels in the LUT.

Detail of Adaptive Mapping (spherical direction)

- Corner case
  - When the sun is below a certain elevation angle
  - We switched to normal latitude-longitude mapping
    - * with non-linear transformations



Sun Elevation Angle | 3 [deg] | 1.5 [deg] | 0 [deg]

Unfortunately, there is a corner case to our approach.

When the sun is below a certain elevation angle, it's a corner case of our mapping and the results have artifacts.

Therefore, we decided to switch to normal latitude-longitude mapping in such situations.

Concept Code

```
Vec3 uvToWolrdSpaceDirection(Real u, Real v)
{
    const Real sun_zenith_distance_deg = theta_sun / pi<Real>() * 180.0;
    const bool sun_in_upper_region = sun_zenith_distance_deg <= threshold_theta_sun_deg();

    if (sun_in_upper_region)
    {
        const Real phi = u * pi<Real>();

        // "nonliner tranformations"
        constexpr float a = 0.50f;
        constexpr float b = 0.80f;
        constexpr float c = 0.25f;
        constexpr float d = 0.50f;
        constexpr float p = 2.0f;
        if (v <= a)
        {
            const auto k = (v / a);
            v = c - c * pow(sqrt(1 - k * k), p);
        }
        else if (v <= b)
        {
            const auto k = (v - b) / (b - a);
            v = c + (d - c) * pow(sqrt(1 - k * k), p);
        }
        else
        {
            const auto k = (v - b) / (1 - b);
            v = 1 - (1 - d) * sqrt(1 - k * k);
        }

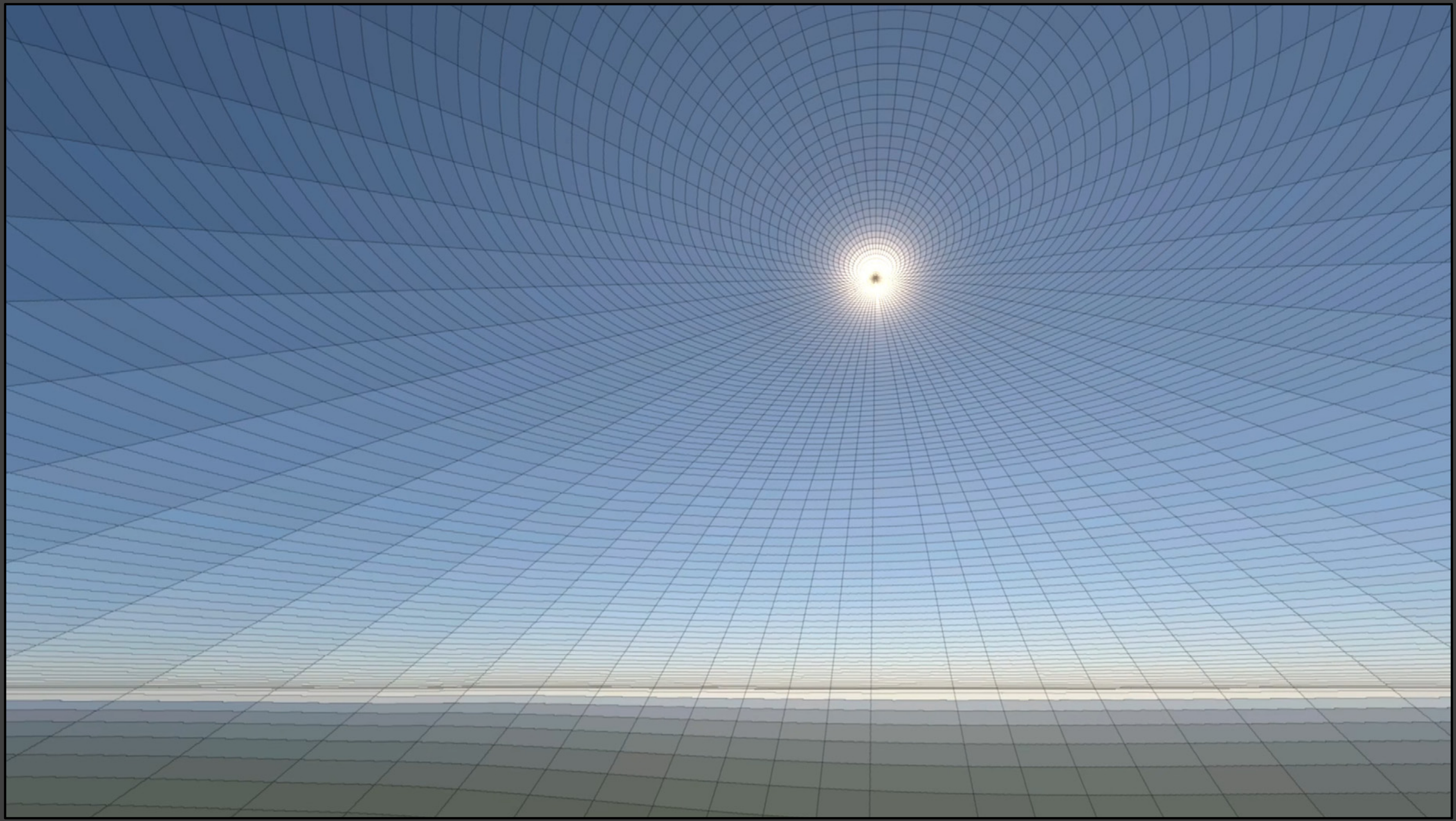
        // "normalization"
        auto edge =
            pi<Real>() / 2 - atan(sin(pi<Real>() / 2 - phi) * tan(theta_sun));
        Real theta = 0;
        if (v <= 0.5)
        {
            theta = v / 0.5 * edge;
        }
        else
        {
            theta = (v / 0.5 – 1.0) * (pi<Real>() - edge) + edge;
        }

        // sun direction is the upward direction
        Vec3 sun_tangent, sun_binormal;
        createOrthoNormalBasis(sun_dir, &sun_tangent, &sun_binormal);
        return polarCoordinateToDirection<Vec3, Real>(theta, phi, sun_dir, sun_tangent, sun_binormal);
    }
    else
    {
        const auto K = std::min(0.5, theta_sun / pi<Real>());

        // "nonlinear transformations"
        if (v <= K)
        {
            const auto k = (v - K) * (v - K);
            v = sqrt(K * K - k);
        }
        else
        {
            const auto k = (v - K) * (v - K);
            v = 1 - sqrt((1 - K) * (1 - K) - k);
        }

        const Real theta = v * pi<Real>();
        const Real phi = u * u * pi<Real>();
        return polarCoordinateToDirection<Vec3, Real>(theta, phi + pi<Real>() / 2);
    }
}
```

Here is the concept code of our nonlinear adaptive mapping.



(video)

This video is an example of LUT visualization, and you can see our adaptive mapping with texel visualization.

As you may have noticed, as the sun elevation changes, the mapping changes adaptively.

When the sun elevation is below the threshold, the entire mapping changes.

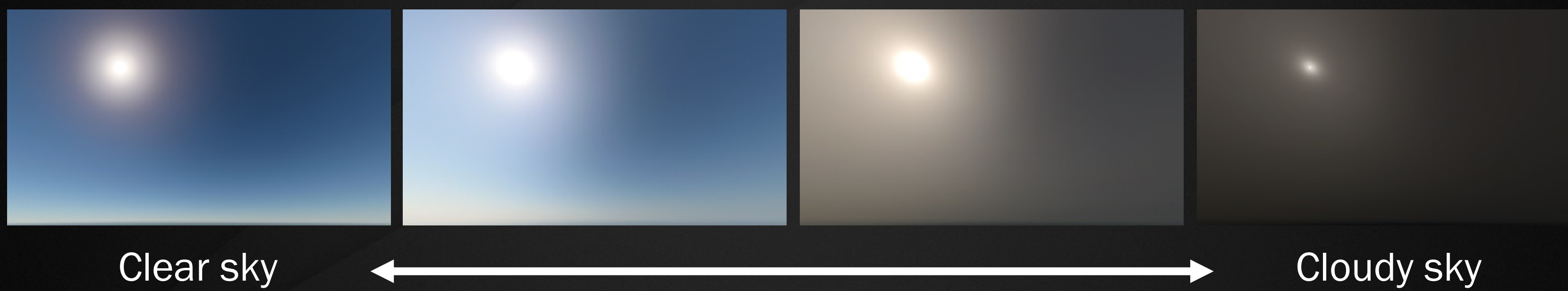
Run-time Sky Rendering Method

Weather Change with Look-up Table

Next, I will talk about weather change with look-up table.

Look-up Table (weather)

- How do we represent weather changes with LUTs?



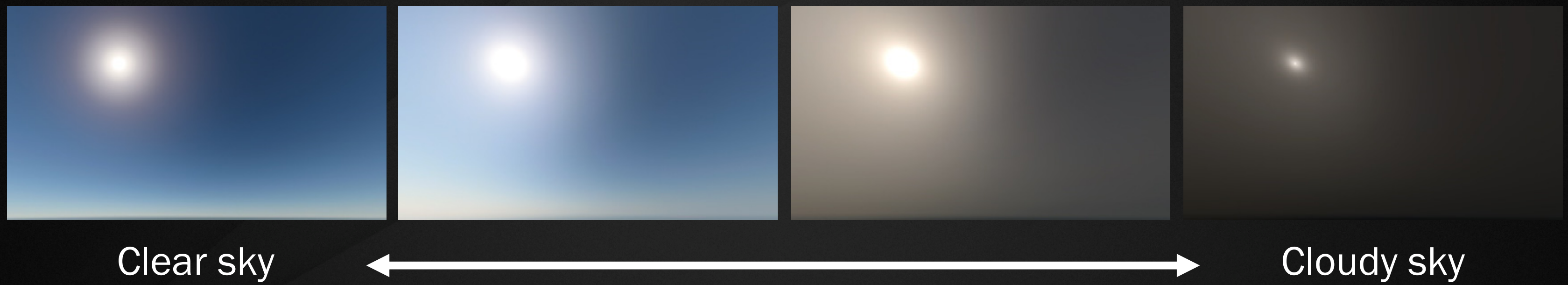
POLYPHONY™ DIGITAL

Now, the sky radiance has been stored in the LUT for time and spherical direction.

The next problem is how to represent changes in weather with LUTs.

Look-up Table (weather)

- How do we represent weather changes with LUTs?
- "Basis Skies"
  - Represents various atmospheric conditions
  - Eight 3D LUT textures in total



In our project, we call the sky corresponding to the various atmospheric conditions "the basis sky".

We use combinations of these basis skies to represent changes from clear skies to cloudy (rainy) skies!

Basis Skies for Weather Change

- Basis skies for clear sky
  - Two clear sky aerosol distributions for two LUTs, “clear“ and “hazed”
  - The aerosol variety is represented by linear interpolation of the two basis skies



“clear”



“hazed”

First, I will explain the basis skies for clear sky.

For the aerosol distribution representing clear skies, here we define only the two extreme distributions, each of which is a LUT.

So the aerosol diversity is represented by linear interpolation of these two bases, “clear” and “hazed”.

Basis Skies for Weather Change

- Basis skies for clear sky
  - U.S. Standard model is used for atmospheric molecules
  - Molecule distribution variations with latitude are ignored

- Simplified to 1-axis control of clear-hazed basis skies

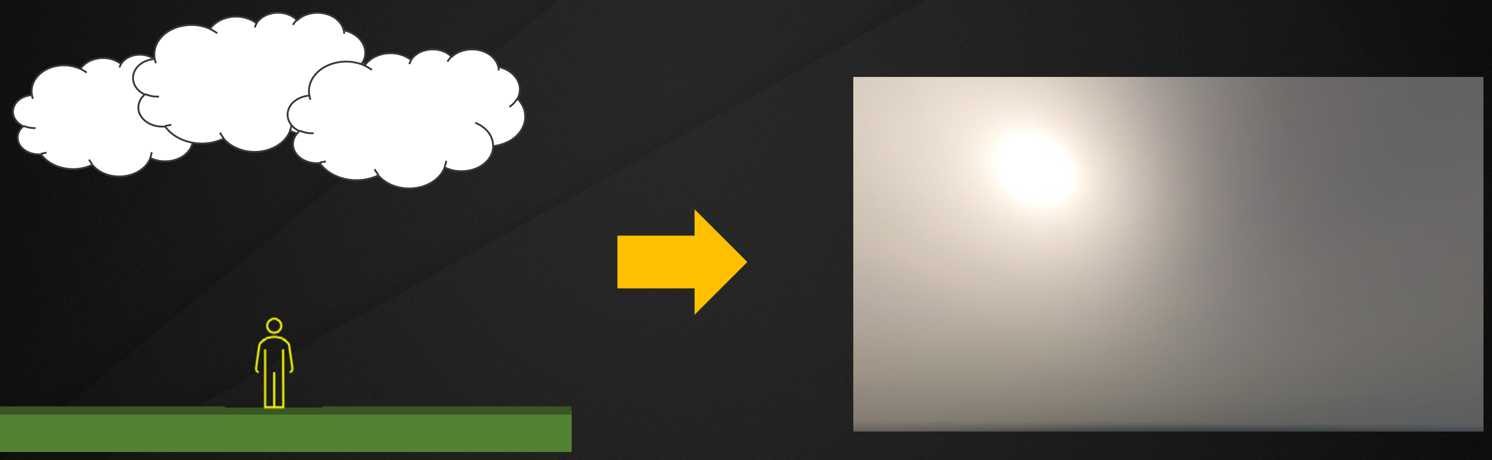
When observation points are limited to the ground, different aerosol conditions often produce similar results (based on experimental results).

Therefore, it is not cost-effective to try to control with many variables, so we decided to use this simple clear-hazed control.

* Each variable is also strongly correlated with each other.

Basis Skies for Weather Change

- Basis skies for cloudy (rainy) sky
  - Huge clouds at high altitudes make the sky uniformly gray, cloudy (rainy) sky



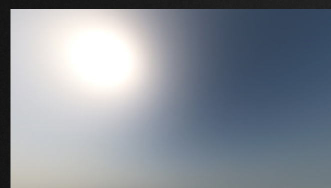
POLYPHONY™ DIGITAL

Next is about the basis skies for cloudy sky.

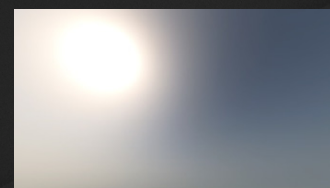
Huge clouds at high altitudes make the sky uniformly gray. This is so called cloudy or rainy sky.

Basis Skies for Weather Change

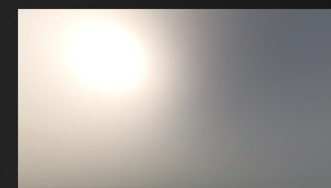
- Basis skies for cloudy (rainy) sky
  - Huge clouds at high altitudes make the sky uniformly gray, cloudy (rainy) sky
  - Six variations of aerosol distributions for LUTs
    - We provide huge clouds as an aerosol distribution for LUTs.



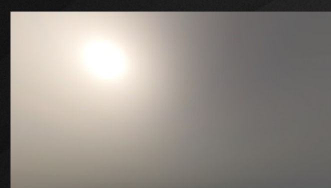
"cloudy0"



"cloudy1"



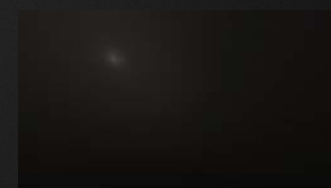
"cloudy2"



"cloudy3"



"cloudy4"



"cloudy5"

POLYPHONY™ DIGITAL

For this purpose, we provide huge clouds as an aerosol distribution of an atmospheric model. These skies are used as basis skies for cloudy sky.

Since the scattering changes rapidly during the formation of high clouds, we prepared 6 basis skies to improve the accuracy of interpolation.

This results in natural weather transitions.

Memory Budget of Look-up Table

- Eight LUT textures in total
  - Sunny: 2 textures
  - Cloudy: 6 textures
- Resolution
  - Sunny: 32 px * 128 px * 64 slices
  - Cloudy: 16 px * 64 px * 64 slices
- Total memory
  - 3.5 MiB

Here is our memory budget for our look-up tables.

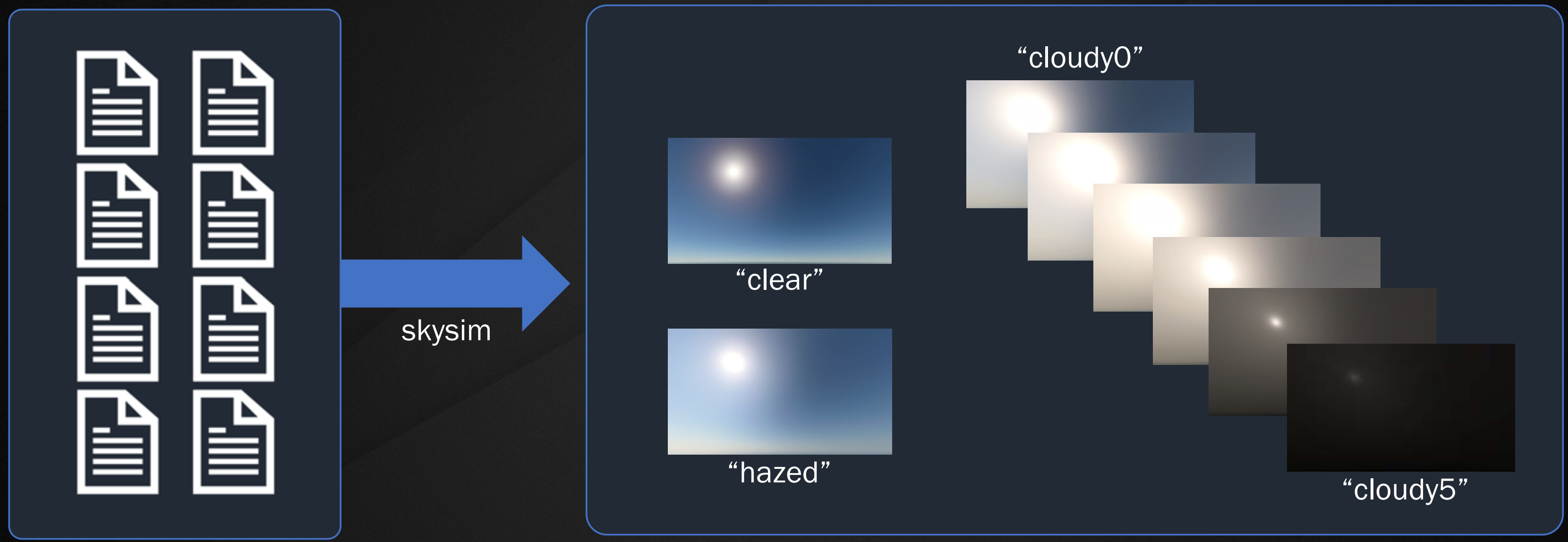
We have a total of eight LUT textures at runtime. Two for sunny and six for cloudy.

The resolution for sunny is 32 px by 128 px with 64 slices. For cloudy, 16 px by 64 px with 64 slices.

The total memory budget is 3.5 MiB.

Interpolation of Basis Skies

- Basis sky Look-up Tables were obtained from each atmosphere models using skysim



Atmosphere model data of basis skies

Look-up Tables of basis skies

Once the atmosphere model data for sunny weather and cloudy weather were obtained, the results of those renderings were converted to LUTs.

Interpolation of Basis Skies

- Weather parameters to interpolate
  - Haziness
  - Cloudiness

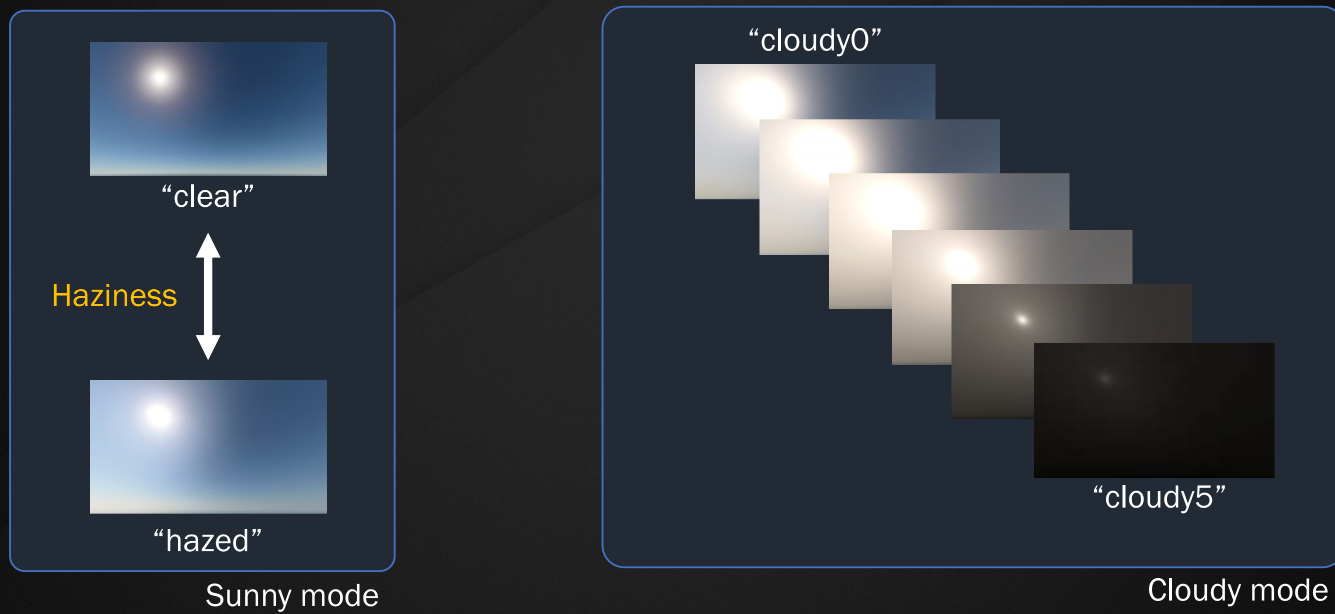


The resulting basis sky LUTs are interpolated using the weather parameters Haziness and Cloudiness.

The weather parameters come from a separately implemented weather simulation.

Interpolation of Basis Skies

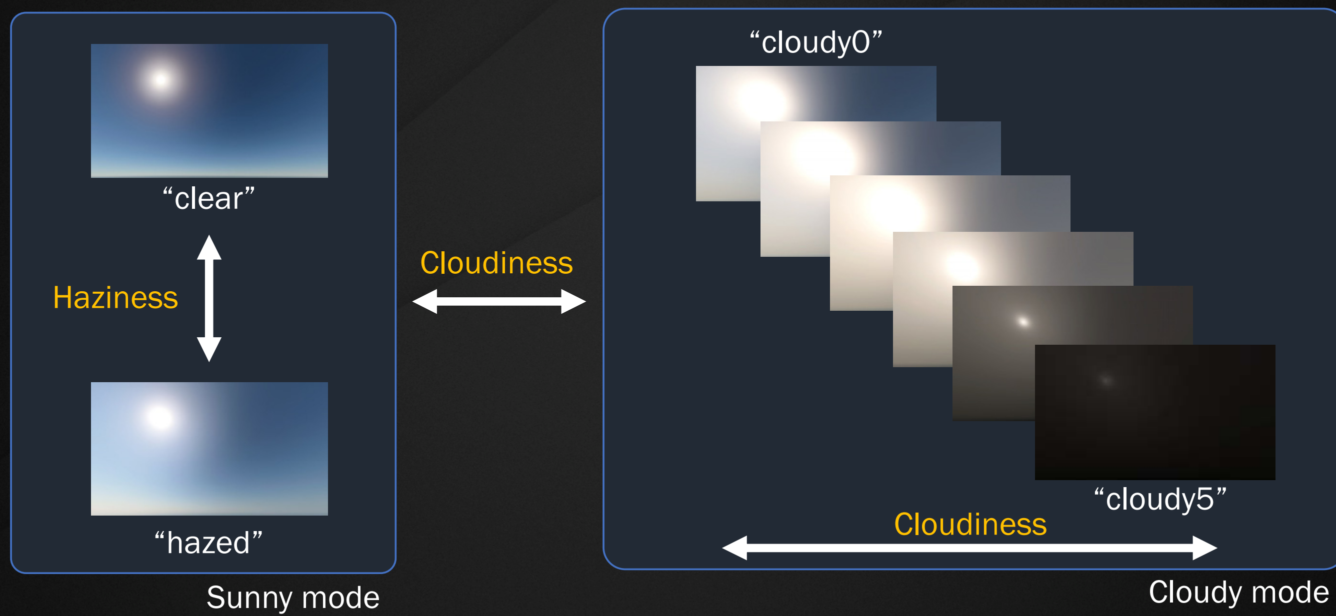
- Weather parameters to interpolate
  - Haziness: Weight for “clear” and “hazed”
  - Cloudiness



Haziness is used to interpolate between "clear" and "hazed".

Interpolation of Basis Skies

- Weather parameters to interpolate
  - Haziness: Weight for “clear” and “hazed”
  - Cloudiness: Weight for “cloudy” skies and mode transition



Cloudiness is used to interpolate between the "cloudy" look-up table. It is also used for the transition between sunny and cloudy mode.



(video)

This is an example result of our weather transition with clouds.

As you can see, the weather transition between clear and cloudy skies is expressed very well.

Other Look-up Tables

- In addition to the sky LUTs, the following LUTs were calculated and used for various processes
  - Fog (two types)
  - Transmittance function
  - Sunlight (parallel light source) intensity
  - Sky average ambient lighting
- The use of LUTs has reduced the computational cost at runtime
  - This is especially important for PS4

Now, we have introduced the sky Look-up Table.

In addition to the sky, we have also used Look-up Tables for fog, transmittance function, sunlight, and ambient lighting.

By using these Look-up Tables, we have greatly reduced the computational cost at runtime.

This is especially important for PS4 due to its lower performance.

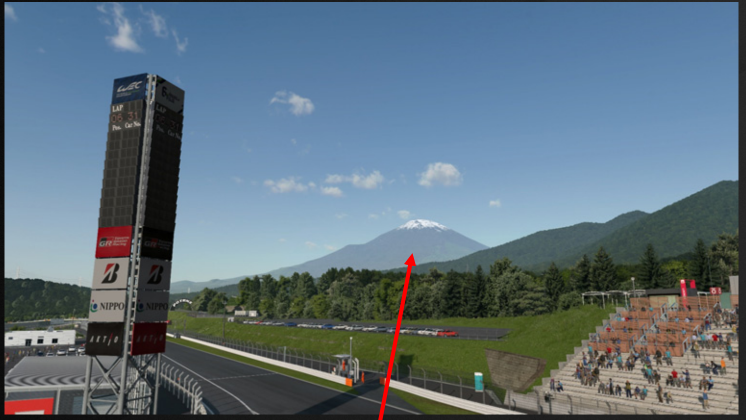
Run-time Sky Rendering Method

Fog

In this section, I want to explain how to introduce fog as part of the sky.

Fog

- Global fog and local fog
  - Global fog from the atmosphere
  - Local fog from local water vapor (volumetric fog)



Global fog



Local fog

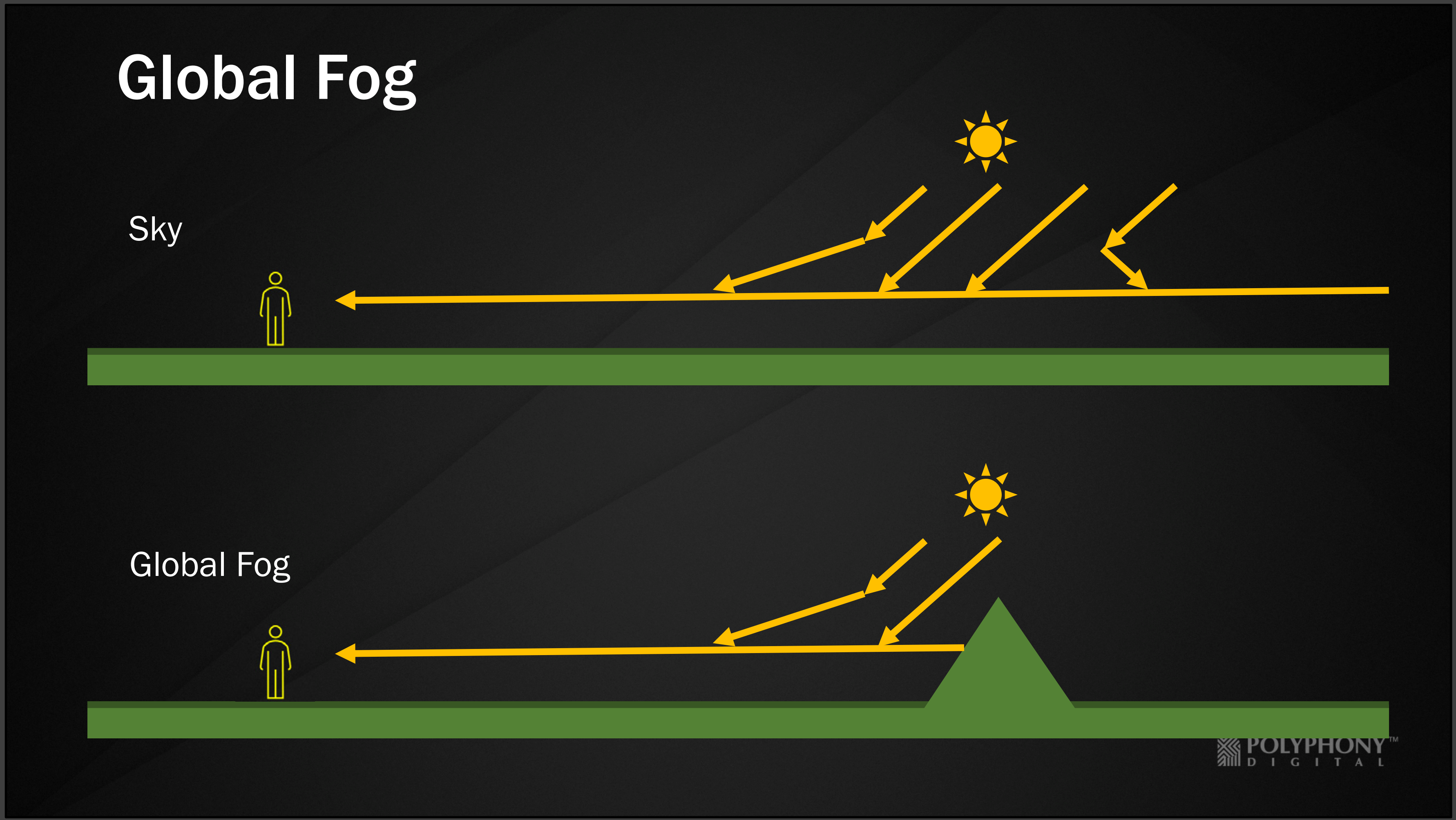
There are two types of fog: global fog and local fog in our game.

Global fog comes from the atmosphere. Local fog comes from local water vapor.

But, I want to focus on the explanation of global fog.



This is an example of our global fog. In the background you can see blue mountains due to the fog.



Global fog can be calculated as atmospheric scattering as in the sky case.

Global fog is consistent with the atmospheric model of the sky and has the same radiance at infinity as the sky.

However, unlike the sky, the calculation ends in the foreground.

Look-up Table for Global Fog

- Scattering components LUT “Fog LUT”
  - Resolution: 32 * 32 * 64 slices

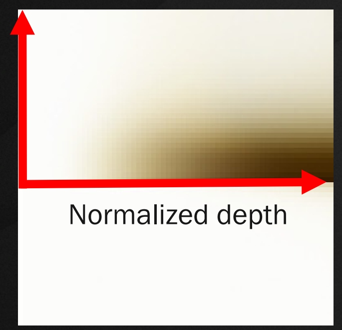
To achieve this, we decided to use a LUT to store the scattering components as in the sky case.

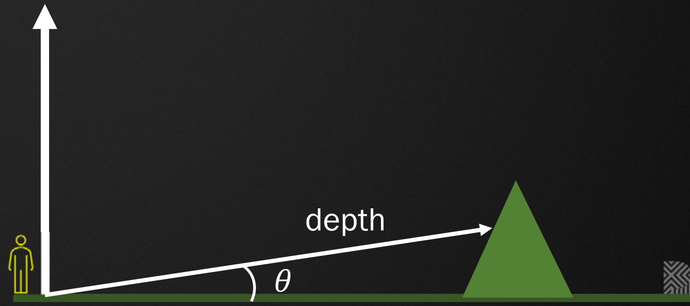
Look-up Table for Global Fog

- Scattering components LUT “Fog LUT”
  - Resolution: 32 * 32 * 64 slices
- Attenuation LUT
  - Resolution: 64 * 64
  - uv are nonlinearly transformed
    - u = pow(normalized depth, 1/3)
    - v = pow(normalized theta, 1/2)

Normalized depth = depth / depth_max
* depth_max = distance to atmosphere boundary

Normalized theta





POLYPHONY DIGITAL

We also decided to use an attenuation LUT for the far field attenuation.

These were all pre-calculated using skysim. In addition, a total of eight sets of separate LUTs were created for each basis sky.

Global Fog Approximation

- Reduction of depth dimension for Scattering LUT
  - We approximate the depth variation with an analytical function

Essentially, the scattering components LUT needs a depth dimension. This is because the depth to the target is used in the fog calculation.

However, this would make the LUT too large and should be avoided in our case.

So we decided to approximate the depth variation with an analytical function.

This means that we don't use the depth dimension for the LUT.

Global Fog Approximation

- Reduction of depth dimension for Scattering LUT
  - We approximate the depth variation with an analytical function

- Approximations
  - a and p are fitted using the reference fog results

$$Fog = LUT(\overrightarrow{dir}) \cdot \frac{1 - \exp(-a \cdot \theta^p \cdot x)}{1 - \exp(-a \cdot \theta^p)}$$

$$x = \text{normalized depth}$$
$$\theta = \text{normalized theta}$$

This equation is used to express the change in fog as a function of the elevation angle of the view and the depth to the object.

Parameters “a” and “p” were fitted using reference fog results for all directions and depths.

Global Fog Problem (1)

- Large error near the camera ☹
  - Fog LUT was calculated including far away
  - Fog is important near the camera



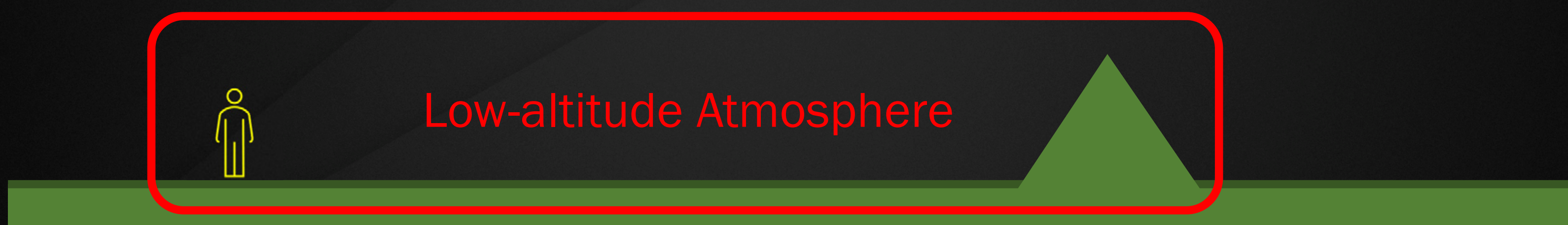
POLYPHONY™ DIGITAL

At first, we simply used the same Look-up Table as the sky for fog calculation, but a problem happened.

The LUT was calculated including too far away, and the error in the neighborhood became large. This is a problem because the fog is important near the camera.

Global Fog Problem (1)

- Large error near the camera ☹
  - Fog LUT was calculated including far away
  - Fog is important near the camera
- Independent fog LUT is better to use ☺
  - Only low-altitude atmosphere affects our global fog, unlike the sky



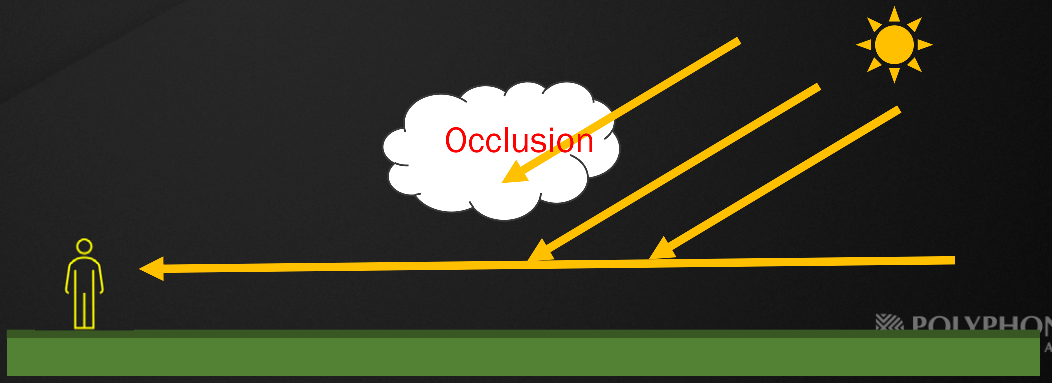
POLYPHONY™ DIGITAL

So we focused that the necessary range for fog only existed at low altitudes below the clouds, and decided to independently calculate a fog LUT that only considers the low-altitude atmosphere.

This greatly reduces the error.

Global Fog Problem (2)

- Large error due to occlusion ☹
  - Occlusion by clouds

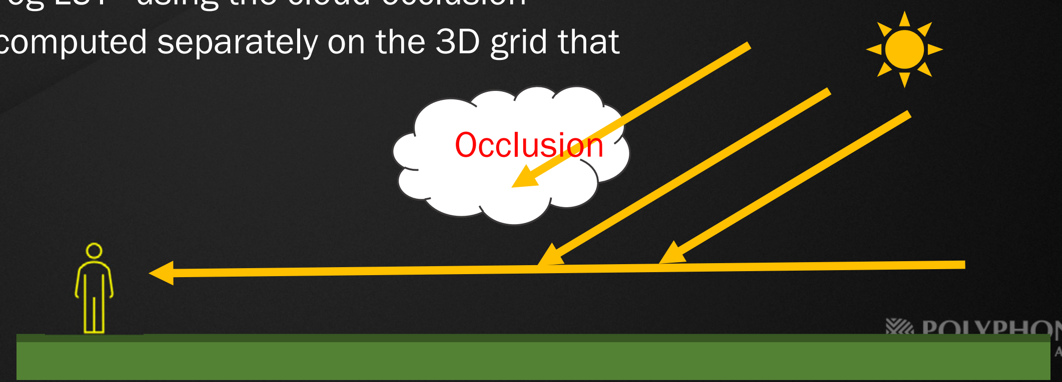


The LUT for fog did not consider the effects of cloud during calculation, resulting in large errors when occlusion was strong.

Therefore, we created an additional LUT that considers the effects of occlusion.

Global Fog Problem (2)

- Large error due to occlusion ☹
  - Occlusion by clouds
- Additional LUT ☺
  - Approximate consideration of occlusion by clouds
    - Fog LUT without single-scattering component
  - "Shadowed Fog LUT"
    - Interpolated with "Fog LUT" using the cloud occlusion
    - Cloud occlusion is computed separately on the 3D grid that covers the world



This LUT is a fog LUT without the single-scattering component due to the sun.

This allows an approximate reduction of the effect of occlusion. We call this additional LUT "Shadowed Fog LUT".

At runtime, the Fog LUT and the Shadowed Fog LUT are interpolated and used based on the cloud occlusion.

This cloud occlusion is computed separately on the 3D grid that covers the world.



This is a in-game result.

As you can see, you can recognize the streaks of light coming through the clouds. This is the effect obtained by the fog I just described.



This is another example.

Agenda

- Introduction

- Physics of Sky
- Simulation of Sky
- Real Earth Atmosphere (and Simulations)
- Skysim --- Our Sky Renderer

- Run-time Sky Rendering Method
- Run-time Cloud Rendering Method
- Summary

POLYPHONY™ DIGITAL

Run-time Cloud Rendering Method

Next I will talk about cloud rendering, specifically cloud modeling and lighting.

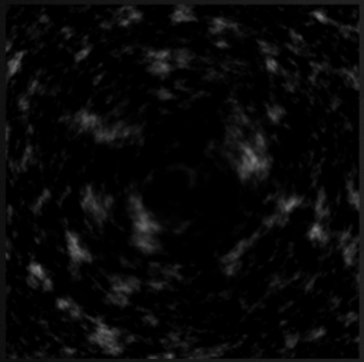
Run-time Cloud Rendering Method

Modeling of Clouds

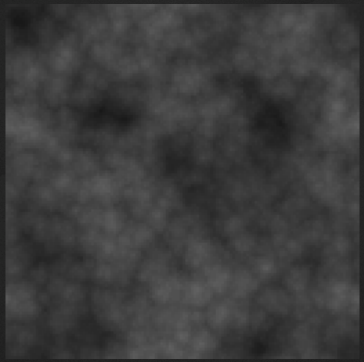
First, let me discuss cloud modeling.

Modeling of Clouds

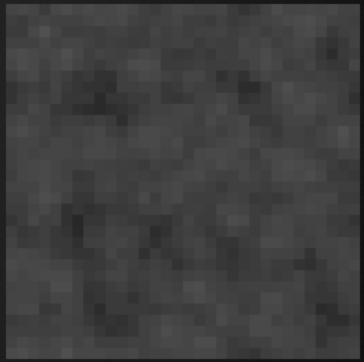
- Raymarching based cumulus clouds
  - Mainly based on [Schneider 2015, 2017]
- Cloud density is computed with three types of noise textures



Cumulus map (2D)



Low frequency noise (3D)



High-frequency noise (3D)

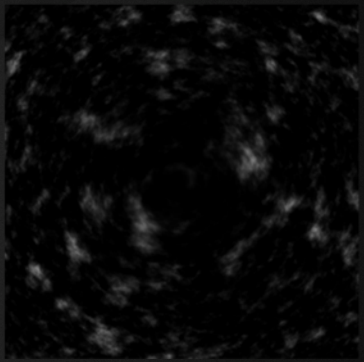
POLYPHONY™ DIGITAL

In reality there are many different types of clouds, but this time we will only model cumulus clouds using ray-marching.

Cumulus clouds are modelled by combining the three types of noise textures.

Modeling of Clouds

- Cumulus map
  - 2D texture (256 * 256)
  - 240 km * 240 km
  - Nonlinear texel distribution optimizations
- Supplied by weather sim
  - Represents the distribution of clouds over the entire world
  - Advected by the wind



Cumulus map (2D)

POLYPHONY™ DIGITAL

The basis for the cloud modeling is the cumulus map, a two-dimensional texture.

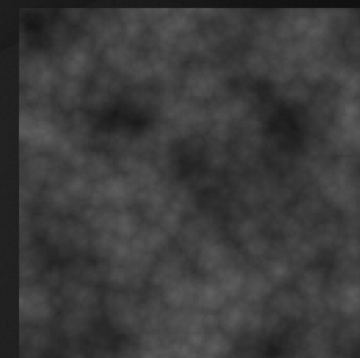
It is a 240 km square noise generated by a separately implemented weather system and is based on Perlin Noise.

This represents the distribution of clouds over the entire world.

The resolution is non-linearly optimized. The more central the world, the more texels are assigned.

Modeling of Clouds

- Low frequency noise
  - 3D texture (128 * 128 * 128)
  - About 5 km scale
  - Perlin-inverted Worley noise
- Texture values are carefully tuned
  - Outline of small clouds mainly depend on this noise
  - Normalized (average value is zero)



Low frequency noise (3D)

The cloud shapes are represented by a low-frequency 3D noise texture.

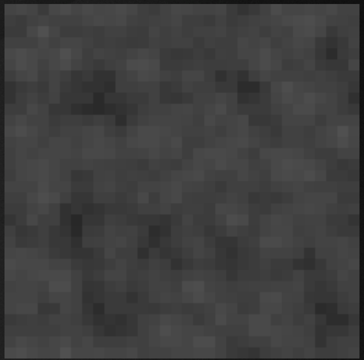
This noise is Perlin-inverted Worley Noise with a resolution of 128, and adds fuzzy cloud impressions due to turbulence.

The outline of small clouds depend mainly on this noise. Therefore, the frequency response and blending are finely tuned.

In addition to that, we normalized the texture so that the average value is zero. By this normalization, the amount of clouds does not vary with the intensity of the noise.

Modeling of Clouds

- High frequency noise
  - 3D texture (32 * 32 *32)
  - About 300 m scale
  - Worley noise
- Adds small detail near the cloud edge
  - Texture is adjusted to reduce buffer aliasing
  - Scale is set several times larger than in reality



High-frequency noise (3D)

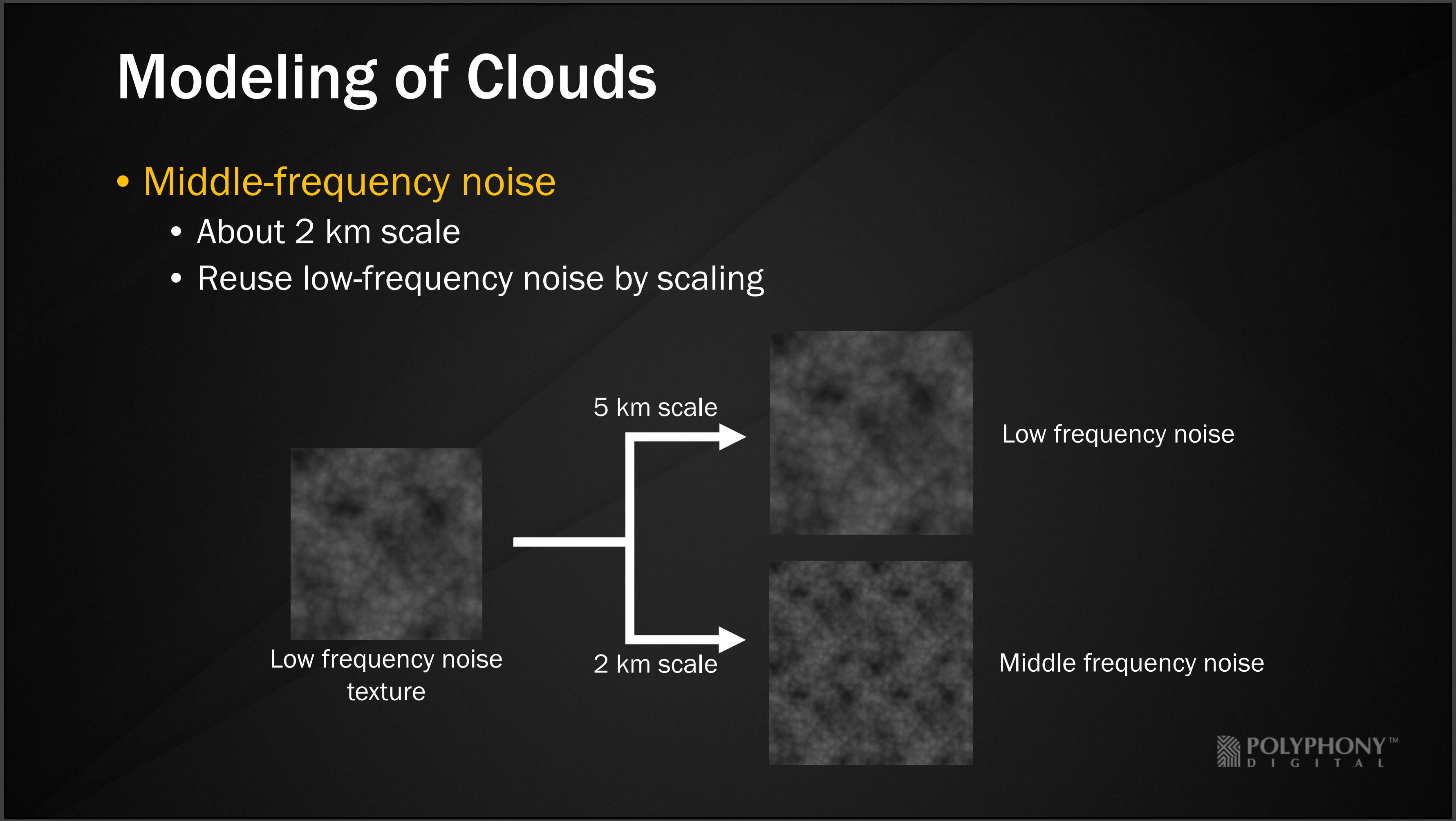
POLYPHONY™ DIGITAL

This is a high frequency noise texture to add high frequency detail.

This noise is Worley noise with a resolution of 32, and adds detail near the cloud edges.

This texture is adjusted to reduce buffer aliasing.

The size scale is set several times larger than in reality. It’s because we want to ensure that the low frequency noise and this noise are not too far apart in frequency space.



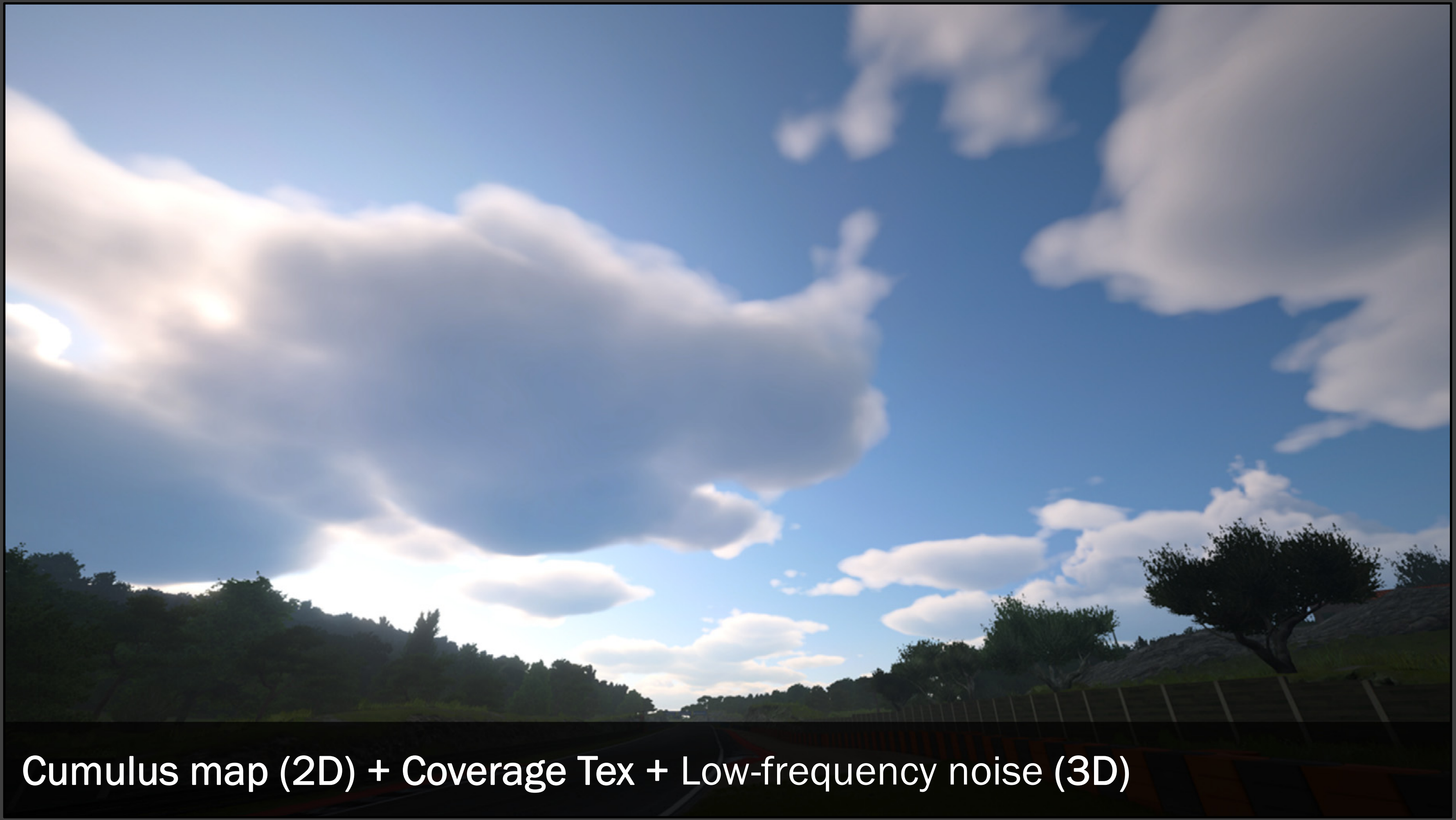
The low and high frequencies have already been described.

However, the mid-frequency component was missing, so we decided to compensate by reusing the low-frequency texture.

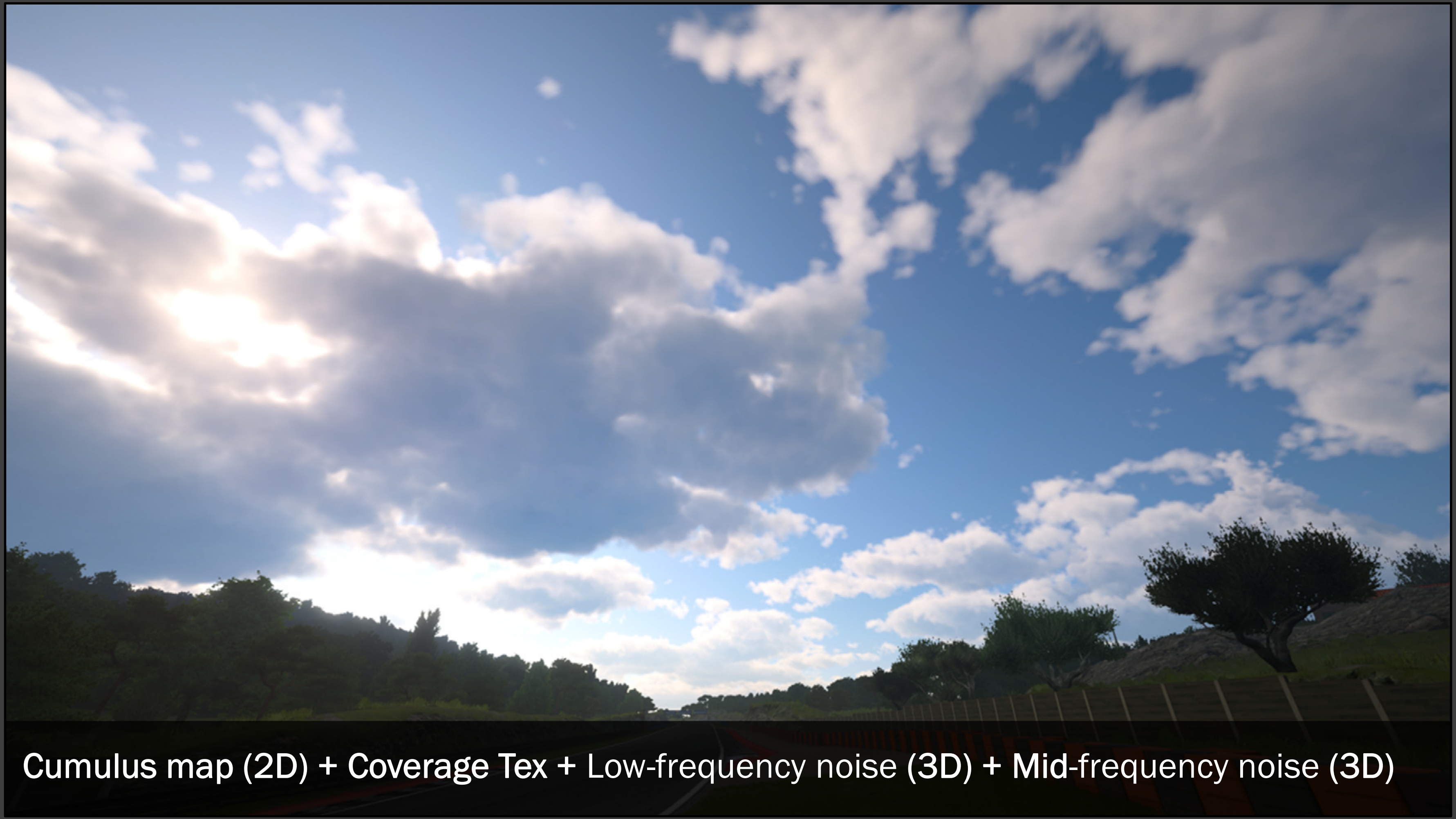


This is a breakdown of our cloud modeling.

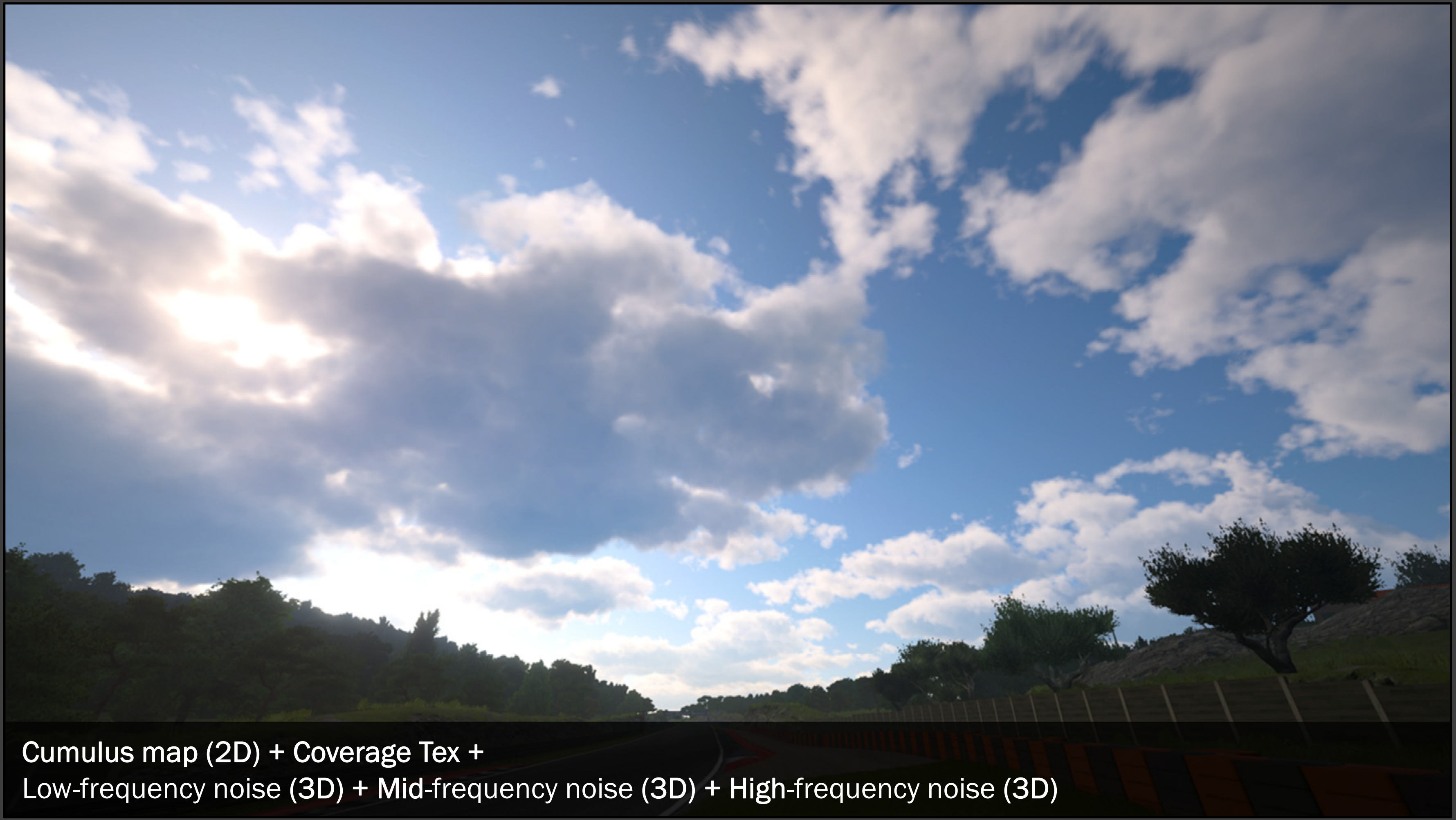
First, two-dimensional cumulus map from our weather system with coverage texture.



With low-frequency noise.



With middle-frequency noise.



Finally, with high-frequency noise.

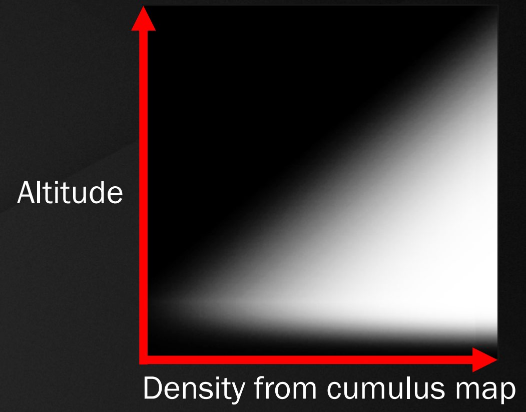


You can compare images with or without noise textures.

Coverage Texture

- 2D Texture
  - Resolution: 256 * 128
  - Hand crafted values

- Determines final cloud density in 3D space
  - Fetch values according to 2D density and altitude, then combine with frequency components



Coverage texture (2D)

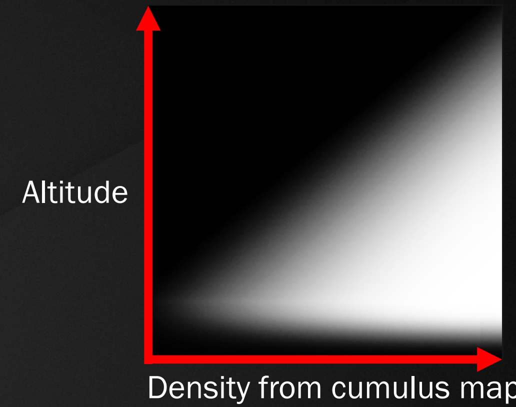
The Coverage Texture is an additional texture for determining three-dimensional cloud density functions.

The horizontal axis corresponds to the density derived from the cumulus map and the vertical axis corresponds to the altitude.

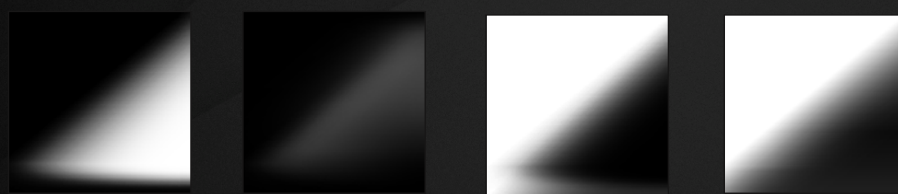
This value is combined with frequency noise to produce the final cloud density field in world space.

Coverage Texture

- 2D Texture
  - Resolution: 256 * 128
  - Hand crafted values
- Determines final cloud density in 3D space
  - Fetch values according to 2D density and altitude, then combine with frequency components
- Following 4 components are packed into RGBA



Coverage texture (2D)



Height modulation | Low-frequency | Mid-frequency | Ambient lighting

4 components are packed into RGBA.

Height modulation component for computing overall density.

Low-frequency and Middle-frequency components for noise textures.

And ambient lighting component.

Run-time Cloud Rendering Method

Lighting of Clouds

Next, let me discuss cloud lighting.

Lighting of Clouds

- Scattering computation by ray-marching
  - Cloud density is computed with described method
  - Our ray-marching is similar to the existing methods

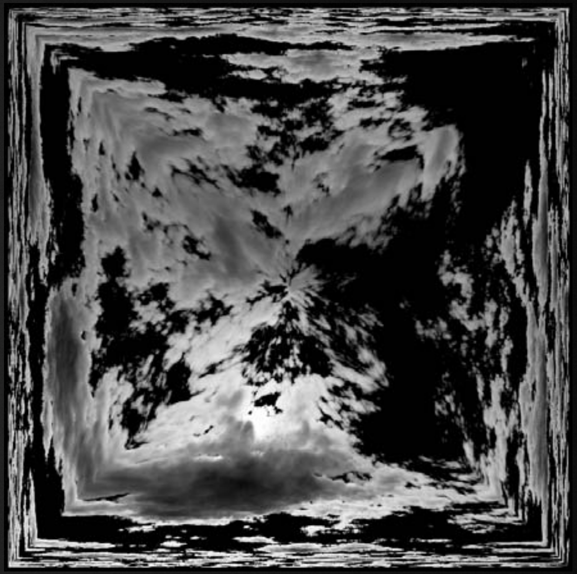


We calculate cloud scattering by ray-marching in the same way as existing methods.

When ray-marching, the cloud density is modeled using the method already explained.

Lighting of Clouds

- "Cloud G-buffer"
  - Origin is the center of the world
  - Render the upper-hemisphere
- Nonlinear texel distribution optimizations
  - More texels are assigned near the horizon



Cloud G-buffer
(upper-hemisphere)

POLYPHONY™ DIGITAL

Our ray-marching results are stored in "cloud G-buffer".

The origin of the ray-marching is the center of the world, then the upper hemisphere is rendered from the origin.

Here, we optimize the texel distributions so that more texels are assigned near the horizon.

Lighting of Clouds

- Nonlinear texel distribution optimizations code

```
float2
gbufferUVtoPolarCoordinates(float u, float v)
{
    u -= 0.5f;
    v -= 0.5f;
    const float phi = atan2(v, u);

    const float R = 0.5f / cos(fmod(phi + PI/4 + 2*PI, PI/2) - PI/4);
    const float x01 = sqrt(length(float2(u, v)) / R);
    const float theta = x01 * (PI/2);

    return float2(theta, phi);
}
```

POLYPHONY™ DIGITAL

This is our nonlinear texel distribution code.

Lighting of Clouds

- Wanted to render independently of cameras
  - Main screen
  - Rearview Mirror
  - Cube Map
  - Reflection surface
  - etc.



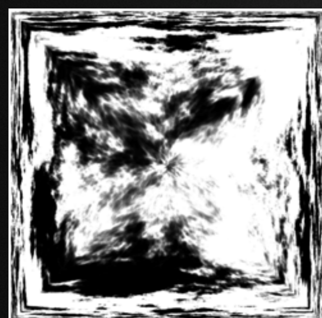
POLYPHONY™ DIGITAL

The reason for rendering in the upper-hemisphere is that we wanted to make it independent of cameras.

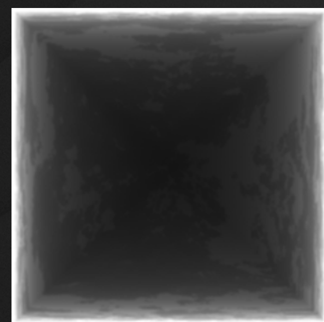
This is because our game renders the sky from various perspectives, such as rear-view mirrors, cube maps, reflection surfaces, etc., in addition to the main screen.

Lighting of Clouds

- "G-buffer" components
  - Opacity to the background sky
  - Distance to the clouds
  - Lighting (in-scattering) by the sun
  - Ambient lighting



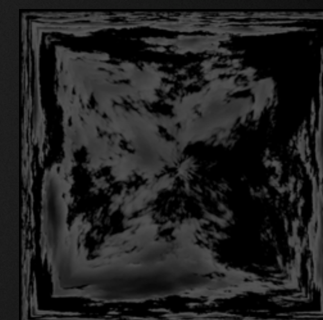
Opacity to
the background (sky)



Distance to clouds



Lighting by the Sun
(in-scatter)



Ambient Lighting

These are the components stored in G-buffer.

Opacity to the background sky

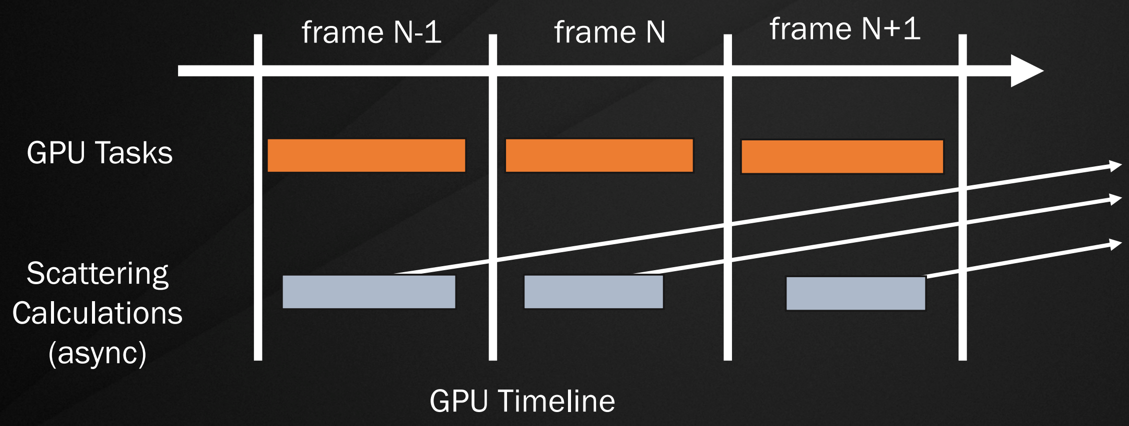
Distance to the clouds

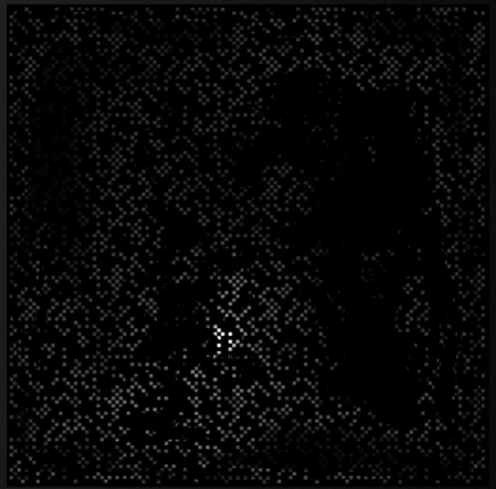
Lighting results by the sun

And Ambient lighting

Motivation of “Cloud G-buffer”

- Multiples frame calculations
  - Divide the scattering calculations over multiple frames.
  - Async compute to do in parallel





(video)

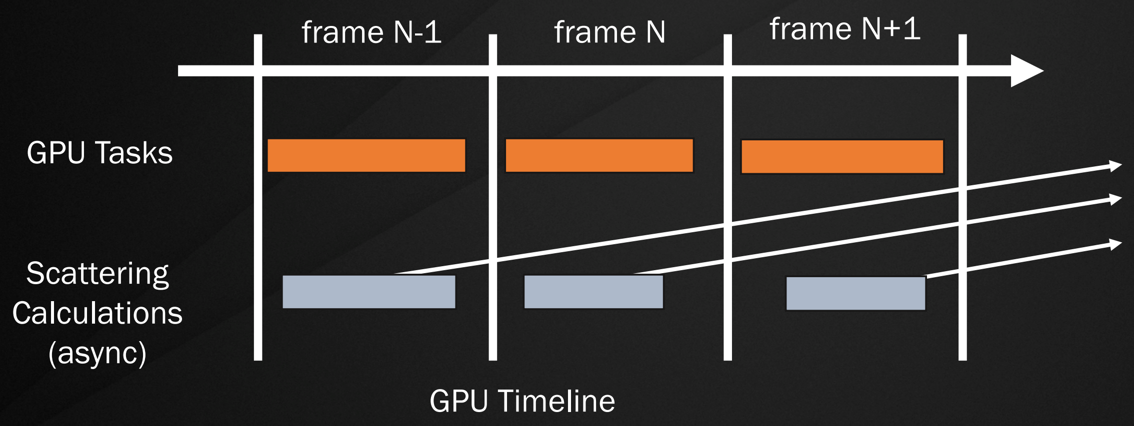
Scattering calculations are computationally expensive. So, we reduce the computational cost per frame by dividing the scattering calculation over multiple frames.

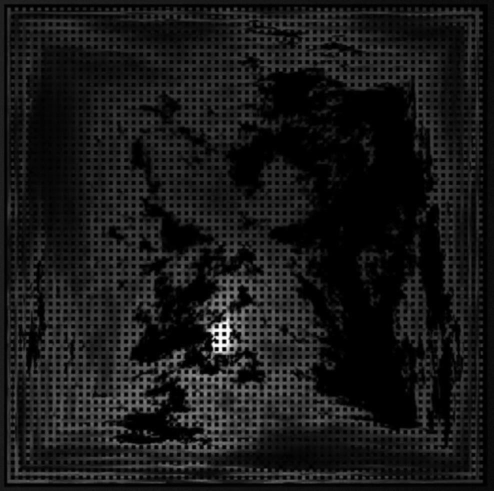
We use async compute to do this in parallel with other processing.

Motivation of “Cloud G-buffer”

- Multiples frame calculations
  - Divide the scattering calculations over multiple frames.
  - Async compute to do in parallel

⇒ Latest calculation results are not always available



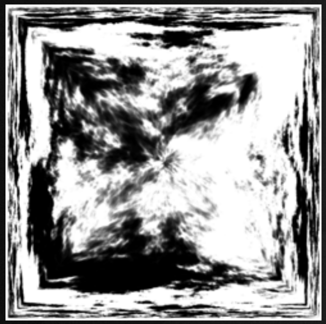


(video)

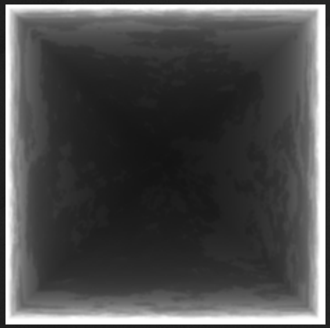
This dividing means that the latest calculation results are not always available.

Motivation of "Cloud G-buffer"

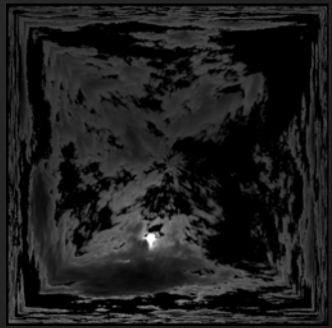
- "G-buffer" to separate lighting and scattering calculation
  - The lighting of the sun and sky can change from frame to frame.
  - Lighting information and scattering results are separated as a "cloud G-buffer" to apply "deferred shading"



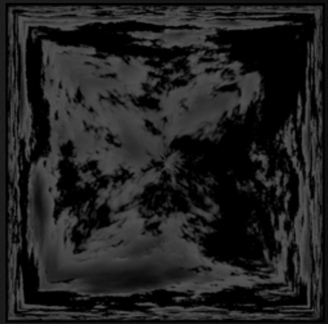
Opacity to
the background (Sky)



Distance to clouds



Lighting by the Sun
(in-scatter)



Ambient Lighting

POLYPHONY™ DIGITAL

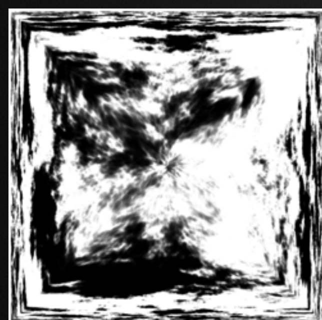
On the other hand, the lighting of the sun and sky can change from frame to frame.

Therefore, the lighting source information and the scattering results are separated and stored as a G-buffer to apply "deferred shading".

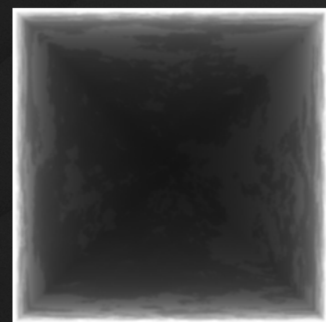
Motivation of "Cloud G-buffer"

- "G-buffer" to separate lighting and scattering calculation
  - The lighting of the sun and sky can change from frame to frame.
  - Lighting information and scattering results are separated as a "cloud G-buffer" to apply "deferred shading"

⇒ The final renderings can change every frame.



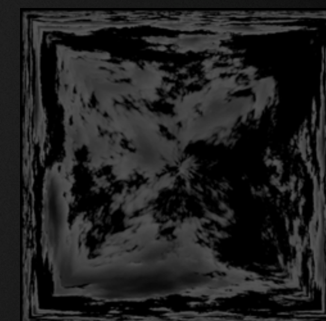
Opacity to
the background (Sky)



Distance to clouds



Lighting by the Sun
(in-scatter)



Ambient Lighting

This allows the lighting to change every frame, and the final rendering to change every frame, too.

This is our main reason to use G-Buffer like this.

Lighting Calculations of G-Buffer

- Multiple-scattering is an ad-hoc approximation based on [Wrenninge et al. 2013]



Single-scattering



Multiple-scattering

POLYPHONY™ DIGITAL

I will explain some of the details of lighting calculations of G-Buffer.

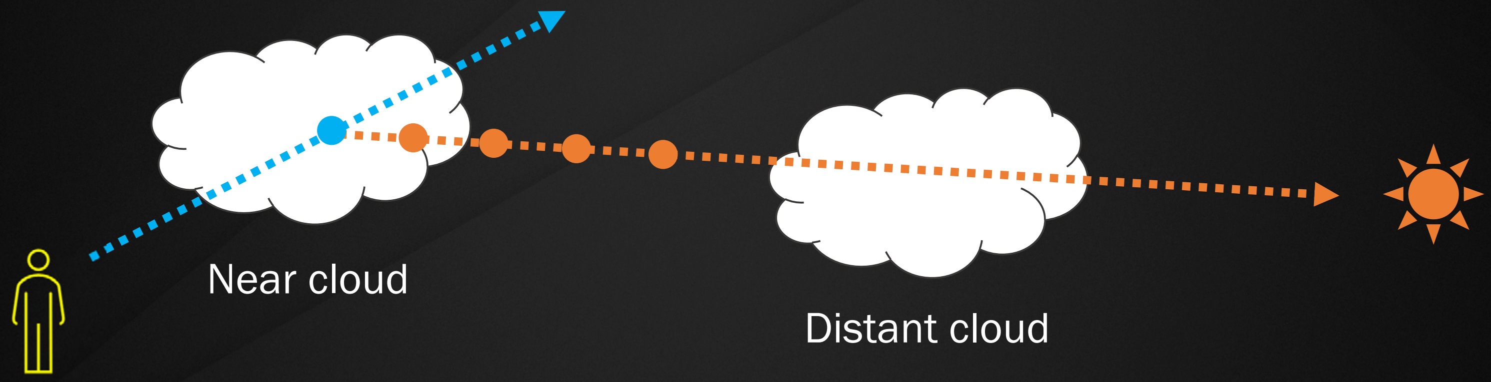
First, about multiple-scattering. As already mentioned, the lighting component of G-Buffer is calculated using ray-marching.

Ray-marching can only compute single-scattering components, but we wanted some multiple-scattering components.

So, we use an existing method based ad-hoc approximation for multiple-scattering. This approximation improves the results, as shown on the right of the figure.

Lighting Calculations of G-Buffer

- Shadow from distant clouds
  - Important in evening lighting
  - Density LOD, adaptive sampling, etc.



POLYPHONY™ DIGITAL

Next, about shadow from clouds.

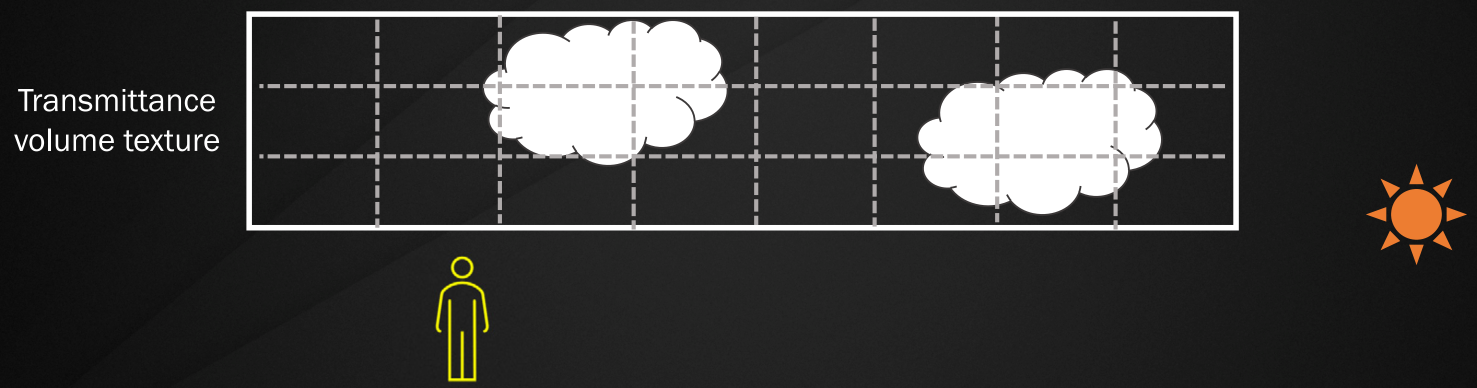
When ray-marching to the sun, we wanted the shadow by clouds, both near and distant, with high accuracy.

This is especially important in evening lighting.   But, this is not easy. Simply doing so will lead to an increase in the number of samples.

There are already ideas such as using density LOD, adaptive sampling, and so on. In our case, we adopted a different approach.

Lighting Calculations of G-Buffer

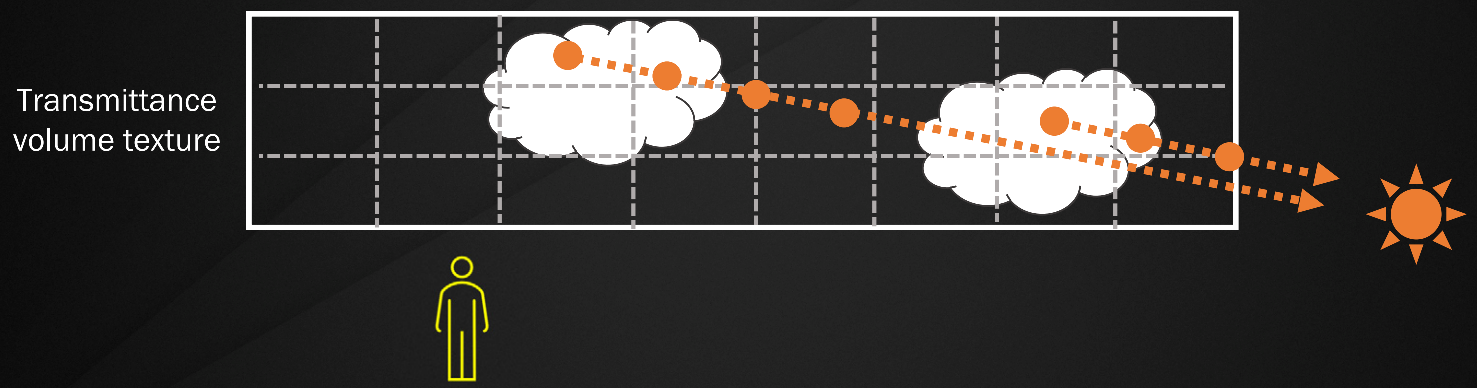
- Transmittance volume texture from distant clouds



We compute a separate 3D volume texture to solve this problem, before calculating cloud G-buffer.

Lighting Calculations of G-Buffer

- Transmittance volume texture from distant clouds
  - Each cell stores transmittance from the sun
  - When computing “cloud G-buffer”, combine ray-marching results and transmittance from distant clouds together



By ray-marching from each cell towards the sun, we store the transmittance from the sun in the volume texture.

This volume texture is used together with the results of ray-marching when calculating cloud G-buffer.

This calculation is also distributed between frames to reduce the cost.

Lighting Calculations of G-Buffer

- Some light leaks are solved with distant cloud shadow



Without distant cloud shadow



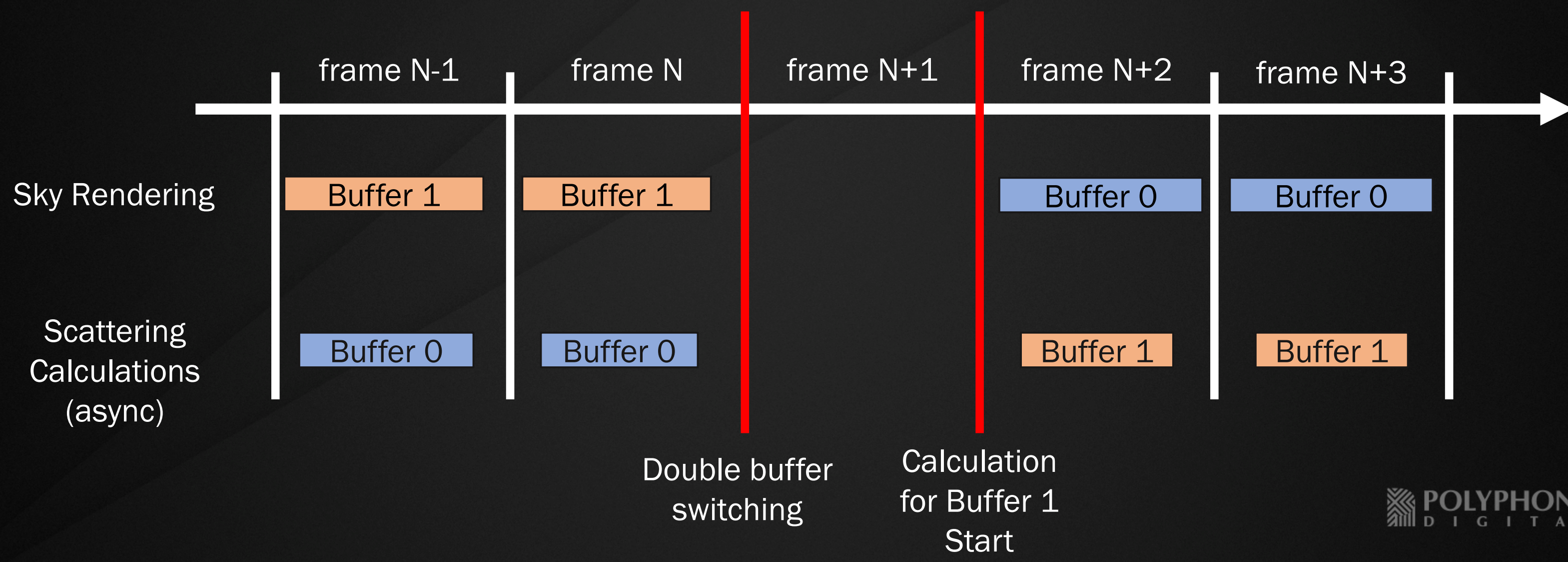
With distant cloud shadow

These are the comparison between with and without the distant cloud shadow.

As you can see in the right image, some light leaks are solved with the distant cloud shadow.

Implementation Detail

- "Cloud G-buffer" is double buffered
  - While the latest buffer is being calculated, previous buffer is used for final sky rendering



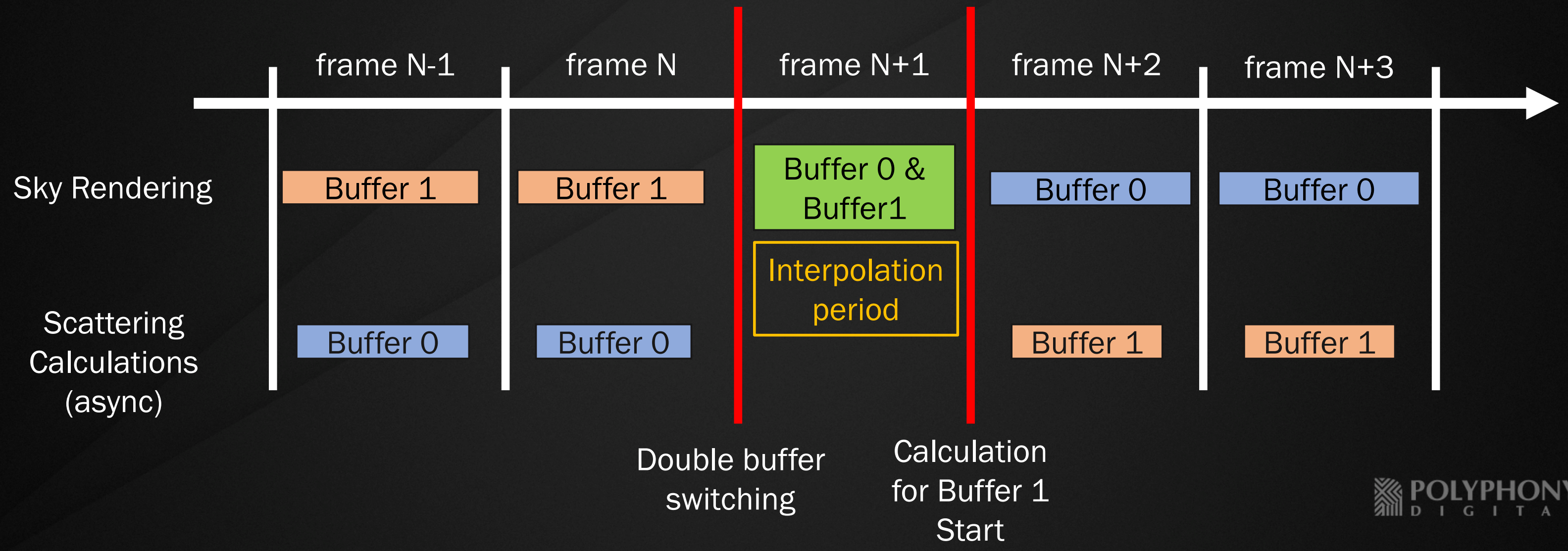
POLYPHONY DIGITAL

From now, let me explain about our implementation detail of cloud G-buffer computations.

Specifically, the G-buffer is double buffered. While the latest buffer is being calculated, the previous buffer is used for the final sky rendering.

Implementation Detail

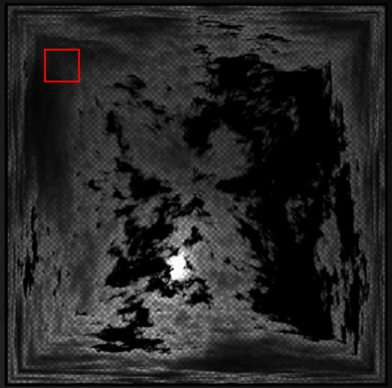
- "Cloud G-buffer" is double buffered
  - While the latest buffer is being calculated, previous buffer is used for final sky rendering
  - Interpolation between two buffers while switching double buffers



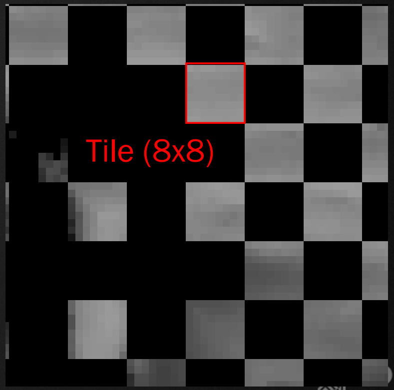
When switching double buffers, interpolation is performed between the two buffers. This makes the switch less noticeable.

Implementation Detail

- Resolution of “cloud G-buffer”
  - PS4: 768 * 768
  - PS5: 1024 * 1024
    - Format is RGBA16F
- The buffer is divided into 8x8 tiles
  - One tile corresponds to one wavefront on GPU



Cloud G-buffer



(video)

The resolution of the buffer is 768x768 on PS4 and 1024x1024 on PS5.

The buffer is divided into 8x8 tiles. A single tile is dispatched as one wavefront.

Implementation Detail

- Number of frames used to calculate single sample of each pixel
  - PS4: 64 frames (lower cost per frame)
  - PS5: 32 frames (higher cost per frame)
- Total computation time is about 6 sec per G-buffer

These are calculation costs and quality on the PS4 and PS5.

The number of frames used to calculate a single sample of each pixel is 64 frames on PS4, 32 frames on PS5.

And the total computation time is 6 seconds.

Implementation Detail

- Number of frames used to calculate single sample of each pixel
  - PS4: 64 frames (lower cost per frame)
  - PS5: 32 frames (higher cost per frame)
- Total computation time is about 6 sec per G-buffer

↓

- Number of samples per pixel (average)
  - PS4: about 5 samples (lower quality G-buffer)
  - PS5: about 10 samples (higher quality G-buffer)

So the number of samples per pixel is about 5 samples on PS4, about 10 samples on PS5.

This means that the computational cost per frame is lower on PS4 and the quality is also lower.

On PS5, the computational cost per frame is higher and the quality is also higher.

Implementation Detail

- Computational cost of “cloud G-buffer”
  - Average computation timing [ms] for one lap of a race track

	Without cloud computation	With cloud computation	Difference
PS4	8.60	9.19	+0.59
PS5	7.86	8.04	+0.18



This is a cost table for GBuffer calculations. The table shows the average load for one lap of a race track.

Our scattering calculation is expensive because of its quality, but the cost per frame is low by dividing the calculation between frames.



(video)

Finally, let me show you the results of our method presented today as a 10x speed video.

As you can see, time and weather are constantly changing.

The background sky is represented by a look-up table at low computational cost.

The cumulus clouds are computed at runtime using ray marching, where the computation is split per frame.

Agenda

- Introduction

- Physics of Sky
- Simulation of Sky
- Real Earth Atmosphere (and Simulations)
- Skysim --- Our Sky Renderer

- Run-time Sky Rendering Method
- Run-time Cloud Rendering Method
- Summary

POLYPHONY™ DIGITAL

Summary

I will conclude today's presentation with a summary.

Summary

- The physics of atmospheric scattering and the methods used to compute it
  - Fog, sky, and clouds can all be rendered by solving the volume rendering equation. There are various ways to give parameters and calculation methods

POLYPHONY™ DIGITAL

First, the physics of atmospheric scattering and its computational methods were introduced.

Fog, sky, and clouds can all be rendered by solving the volume rendering equation. There are various ways to give parameters and calculation methods.

Summary

- The physics of atmospheric scattering and the methods used to compute it
  - Fog, sky, and clouds can all be rendered by solving the volume rendering equation. There are various ways to give parameters and calculation methods
- Offline rendering techniques for high-quality skies
  - Each parameter required for rendering is modeled based on a wide range of research, and offline rendering is performed using our own renderer, "skysim"

POLYPHONY™ DIGITAL

Next, we presented a high-quality off-line sky rendering technique.

Each parameter required for rendering is modeled based on a wide range of research,

and offline rendering is performed using our own renderer, "skysim".

Summary

- The physics of atmospheric scattering and the methods used to compute it
  - Fog, sky, and clouds can all be rendered by solving the volume rendering equation. There are various ways to give parameters and calculation methods
- Offline rendering techniques for high-quality skies
  - Each parameter required for rendering is modeled based on a wide range of research, and offline rendering is performed using our own renderer, "skysim"
- Sky and cloud rendering at runtime
  - High quality and fast sky rendering is achieved by using Look-up Tables. These LUTs are converted from the results of atmospheric scattering simulations
  - Our approach to modeling and lighting of clouds

Finally, sky and cloud rendering using Look-up Table at run-time was presented.

High quality and fast sky rendering is achieved by using Look-up Tables. These LUTs are converted from the results of atmospheric scattering simulation.

I have presented our approach to modeling and lighting of clouds

Thanks!

- Takeshi Yokouchi
- Akihiko Tan
- Atsushi Hayashi
- Shuichi Takano
- Masato Mukoda
- Kohei Ishiyama

- All Polyphony Digital Inc. staffs

POLYPHONY DIGITAL

This is the acknowledgement slide.

And our final slide of today's talk.

Thank you for your kind attention.

References

- [Novák et al. 2018] Jan Novák, Iliyan Georgiev, Johannes Hanika, and Wojciech Jarosz. 2018. Monte Carlo Methods for Volumetric Light Transport Simulation. Computer Graphics Forum (Proceedings of Eurographics - State of the Art Reports) 37, 2 (May 2018).
- [Nishita et al. 1993] Tomoyuki Nishita, Takao Sirai, Katsumi Tadamura, and Eihachiro Nakamae. 1993. Display of the Earth Taking into Account Atmospheric Scattering. In Proceedings of the 20th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '93), Association for Computing Machinery, Anaheim, CA, 175–182. DOI:https://doi.org/10.1145/166117.166140
- [Nishita et al. 1996] Tomoyuki Nishita, Yoshinori Dobashi, and Eihachiro Nakamae. 1996. Display of Clouds Taking into Account Multiple Anisotropic Scattering and Sky Light. In Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '96), Association for Computing Machinery, New York, NY, USA, 379–386. DOI:https://doi.org/10.1145/237170.237277
- [Bruneton and Neyret 2008] Eric Bruneton and Fabrice Neyret. 2008. Precomputed Atmospheric Scattering. Computer Graphics Forum (2008). DOI:https://doi.org/10.1111/j.1467-8659.2008.01245.x
- [Hosek and Wilkie 2012] Lukas Hosek and Alexander Wilkie. 2012. An Analytic Model for Full Spectral Sky-Dome Radiance. ACM Trans. Graph. 31, 4 (July 2012). DOI:https://doi.org/10.1145/2185520.2185591
- [Hillaire 2020] Sébastien Hillaire. 2020. A Scalable and Production Ready Sky and Atmosphere Rendering Technique. Computer Graphics Forum 39, 4 (2020), 13–22. DOI:https://doi.org/https://doi.org/10.1111/cgf.14050
- [Wilkie et al. 2021] Alexander Wilkie, Petr Vevoda, Thomas Bashford-Rogers, Lukáš Hošek, Tomáš Iser, Monika Kolářová, Tobias Rittig, and Jaroslav Křivánek. 2021. A Fitted Radiance and Attenuation Model for Realistic Atmospheres. ACM Trans. Graph. 40, 4 (July 2021). DOI:https://doi.org/10.1145/3450626.3459758

POLYPHONY™ DIGITAL

References

- [Novák 2014] Jan Novák, Andrew Selle, and Wojciech Jarosz. 2014. Residual Ratio Tracking for Estimating Attenuation in Participating Media. ACM Trans. Graph. 33, 6 (November 2014). DOI:https://doi.org/10.1145/2661229.2661292
- [cmfs] Colour & Vision Research Laboratory. http://cvrl.ucl.ac.uk/cmfs.htm
- [Krueger and Minzner 1976] A. Krueger and Raymond Minzner. 1976. A MidLatitude Ozone Model for the 1976 U.S. Standard Atmosphere. Journal of Geophysical Research 81, (August 1976), 4477–4481. DOI:https://doi.org/10.1029/JC081i024p04477
- [Omar et al. 2005] Ali H. Omar, Jae-Gwang Won, David M. Winker, Soon-Chang Yoon, Oleg Dubovik, and M. Patrick McCormick. 2005. Development of global aerosol models using cluster analysis of Aerosol Robotic Network (AERONET) measurements. Journal of Geophysical Research: Atmospheres 110, D10 (2005). DOI:https://doi.org/https://doi.org/10.1029/2004JD004874
- [O3 Spectra] https://www.iup.uni-bremen.de/gruppen/molspec/databases/referencespectra/o3spectra2011/index.html
- [Bohren and Huffman, 1983] Craig Bohren and Donald Huffman. 1998. Absorption and Scattering of Light by Small Particles. Wiley Science Paperback Series.
- [Schneider 2015] Andrew Schneider. 2015. The Real-time Volumetric Cloudscapes of Horizon: Zero Dawn. Advances in Real Time Rendering, Part I. In ACM SIGGRAPH 2015 Courses (SIGGRAPH '15), Association for Computing Machinery, Los Angeles, California. DOI:https://doi.org/10.1145/2776880.2787701

POLYPHONY™ DIGITAL

References

- [Schneider 2017] Andrew Schneider. 2017. Nubis: Authoring Real-Time Volumetric Cloudscapes with the Decima Engine. Advances in Real-Time Rendering, Part I. In ACM SIGGRAPH 2017 Courses (SIGGRAPH ’17), Association for Computing Machinery, Los Angeles, California. DOI:https://doi.org/10.1145/3084873.3096476
- [Hillaire 2016] Sébastien Hillaire. 2016. Physically Based Sky, Atmosphere & Cloud Rendering in Frostbite. Physically Based Shading in Theory and Practice. In ACM SIGGRAPH 2016 Courses (SIGGRAPH ’16), Association for Computing Machinery, Anaheim, California. DOI:https://doi.org/10.1145/2897826.2927353
- [Bauer 2019] Fabian Bauer. 2019. Creating the Atmospheric World of Red Dead Redemption 2: A Complete and Integrated Solution. Advances in Real-Time Rendering in Games, Part 2. In ACM SIGGRAPH 2019 Courses (SIGGRAPH ’19), Association for Computing Machinery, Los Angeles, California. DOI:https://doi.org/10.1145/3305366.3335036
- [Wrenninge et al. 2013] Magnus Wrenninge, Chris Kulla, and Viktor Lundqvist. 2013. Oz: The Great and Volumetric. In ACM SIGGRAPH 2013 Talks (SIGGRAPH ’13), Association for Computing Machinery, Anaheim, California. DOI:https://doi.org/10.1145/2504459.2504518
- [Kider et al. 2014] Joseph T. Kider, Daniel Knowlton, Jeremy Newlin, Yining Karl Li, and Donald P. Greenberg. 2014. A Framework for the Experimental Comparison of Solar and Skydome Illumination. ACM Trans. Graph. 33, 6 (November 2014). DOI:https://doi.org/10.1145/2661229.2661259

POLYPHONY™ DIGITAL